From 8999306208705873093
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,58f35b01f502870a,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-09-26 21:24:37 PST
Path: utzoo!attcan!uunet!wyse!vsi1!ames!pasteur!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i075: achlib -- Memory management and other routines for MadMac
part02/02
Keywords: MadMac
Message-ID: <4946@saturn.ucsc.edu>
Date: 27 Sep 88 04:24:37 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1004
Approved: koreth@ssyx.ucsc.edu

Submitted-by: achowe@watmsg.waterloo.edu (Anthony C. Howe)
Posting-number: Volume 1, Issue 75
Archive-name: achlib/part02

```
        tst.b      (a0)
        beq.s      .X
        move.b     (a0)+,(a1)+
        subq.w     #1,d0
        bne        .1
.X:
        clr.b      (a1)
        rts



;void strcat( char* s1, char* s2 )

         .text
_strcat::
         .cargs        .str1.l, .str2.l
        move.l     .str1(sp), a0
.1:
        tst.b      (a0)+
        bne             .1
        subq.l     #1,a0
        movea.l    .str2(sp),a1
.2:
        move.b     (a1)+,(a0)+
        bne        .2
        rts



;long strlen( char* s )

        .text
```

```
_strlen::
        moveq.l    #-1,d0
        movea.l    4(sp),a0
.1:
        addq.l     #1,d0
         tst.b          (a0)+
        bne        .1
        rts



;*** Find next token in string.
;
; e  long pointer to string / NULL to continue with last string
;    long pointer to delimeter string / NULL use default delimeters
;
; x  long pointer to start of token / NULL if no tokens left
;
;    Note: a) original string will be modified.
;          b) leading spaces are skipped.

          .data
DEFDELIM: .dc.b     SPACE, ',', 0

          .bss
_NEXTTOK::
          .ds.l     1

          .text
_STRTOK::
          move.l    4(sp), d0           ;get string
          bne.s     .1                  ;if NULL then
          move.l    _NEXTTOK, d0        ; continue with previous
          beq.s     .X                  ; string, return NULL if EOS
.1:
          movea.l   d0, a0
.2:
          cmpi.b    #SPACE, (a0)+       ;skip leading spaces
          beq       .2
          subq.l    #1, a0
          move.l    a0, d0              ;start of token to return
          move.l    8(sp), d1           ;get delimeter string
          bne.s     .3                  ;if NULL then use default
          move.l    #DEFDELIM, d1
.3:
          movea.l   d1, a1              ;reset delimeter string
          move.b    (a0)+, d2           ;get char in token
.4:
          cmp.b     (a1), d2            ;check against delims
          beq.s     .5
          tst.b     (a1)+
          bne       .4                  ;end of delim string?
```

```
          bra       .3
.5:
          clr.b     -1(a0)              ;terminate token at delim
          tst.b     d2                  ;was it end of search string
          bne.s     .6
          suba.l    a0, a0              ;save NULL for next search
.6:
          move.l    a0, _NEXTTOK        ;remember current position
.X:
          rts


;*** Find index of character in string
;
; e  long pointer to string
;    word character value
;
; x  D0.L = index into string if character is found else -1

          .text
_FINDI::
          move.l    4(sp),a0       ;pointer to string
          moveq.l   #-1, d0
          move.w    8(sp),d1       ;character
.1:
          tst.b     (a0)           ;EOS?
          beq.s     .X
          cmp.b     (a0)+, d1      ;check character
          bne       .1
          move.l    a0, d0
          sub.l     4(sp), d0      ;current - original
          subq.l    #1, d0         ;adjust overrun -1
.X:
          rts


;*** Find index of last occurence of character in string
;
; e  long pointer to string
;    word character value
;
; x  D0.L = index into string if character is found else -1

          .text
_LFINDI::
          movea.l   4(sp), a0
          moveq.l   #-1, d0
          move.w    8(sp), d1
          bra.s     .2
.1:
          cmp.b     (a0)+, d1
          bne.s     .2
          move.l    a0, d0
```

```
          sub.l     4(sp), d0
          subq.l    #1, d0
.2:
          tst.b     (a0)
          bne       .1
          rts


;*** Deque Rouitines

;*** Make Deque
;
; e  long      # of long words for deque
;
; x  D0.L =    ptr to deque structure / NULL unsuccessful

          .text
_MKDEQ::
          link      a6, #0
          movem.l   a3-a4, -(sp)
          move.l    8(a6), d1
          lsl.l     #2, d1
          addi.l    #16, d1
          move.l    d1, -(sp)
          move.w    #$48, -(sp)
          trap      #1
          addq.l    #6, sp
          tst.l     d0
          beq.s     .X
          movea.l   d0, a0
          movea.l   d0, a1
          adda.l    #16, a1             ;base of deque
          movea.l   a1, a2              ;left pointer
          movea.l   a1, a3              ;right pointer
          movea.l   a1, a4
          adda.l    d1, a4              ;end of deque +1
          movem.l   a1-a4, (a0)         ;save deque structure
.X:
          movem.l   (sp), a3-a4
          unlk      a6
          rts


;*** Remove Deque
;
; e  long      ptr to deque structure
;
; x  D0.W =    NULL for success / error code

          .text
_RMDEQ::
          move.l    4(sp), -(sp)
          move.w    #$49, -(sp)
```

```
          trap      #1
          addq.l    #6, sp
          rts


;*** Add an element to deque
;
; e  long      ptr to deque structure
;    word      0 left / 1 right
;    long      value to store

          .text
_DEQADD::
          link      a6, #0
          movem.l   a3-a4, -(sp)
          movea.l   8(a6), a0           ;ptr to deque structure
          movem.l   (a0), a1-a4         ;fetch deque structure
          move.l    14(a6), d0          ;value
          move.w    12(a6), d1          ;if adding to the right
          bne.s     .2
          cmpa.l    a1, a2
          bhi.s     .1                  ;if begin < left then
          movea.l   a4, a2              ;wrap deque to end
.1:
          move.l    d0, -(a2)           ;save value
          bra.s     .X
.2:
          move.l    d0, (a3)+           ;save value
          cmpa.l    a3, a4              ;if right < end then
          bhi.s     .X
          movea.l   a1, a3              ;wrap deque to beginning
.X:
          movem.l   a1-a4, (a0)         ;update deque structure
          movem.l   (sp), a3-a4
          unlk      a6
          rts


;*** Delete value from deque
;
; e  long      ptr to deque structure
;    word      0 left / 1 right
;
; x  D0.L      value

          .text
_DEQDEL::
          link      a6, #0
          movem.l   a3-a4, -(sp)
          movea.l   8(a6), a0
          movem.l   (a0), a1-a4
          move.w    12(a6), d1
          bne.s     .1
```

```
          move.l    (a2)+, d0           ;get value
          cmpa.l    a2, a4              ;if left < end then
          bhi.s     .X
          movea.l   a1, a2              ;wrap deque to beginning
          bra.s     .X
.1:
          cmpa.l    a1, a3              ;if begin < right then
          bhi.s     .2
          movea.l   a4, a3              ;wrap deque to end
.2:
          move.l    -(a3), d0           ;get value
.X:
          movem.l   a1-a4, (a0)         ;update deque structure
          movem.l   (sp), a3-a4
          unlk      a6
          rts


          .globl    _PUTDEC
          .globl    PUTDEC

;*** Print Unsigned Decimal (0-65535)
;
; _PUTDEC      e    word value
; PUTDEC       e    D0.W value
;              x    none

          .text
_PUTDEC:
          move.w    4(sp),d0
PUTDEC:
          ext.l     d0
          divu      #10,d0
          beq.s     PUTDEC1
          move.l    d0,-(sp)
          bsr       PUTDEC
          move.l    (sp)+,d0
PUTDEC1:
          swap      d0
          addi.w    #$30,d0
          move.w    d0,-(sp)
          move.w    #CCONOUT,-(sp)
          trap      #1
          addq.l    #4,sp
          rts


;*** Print Unsigned Decimal
;
; _PUTLONG     e    long value
; PUTLONG      e    D0.L value
;              x    none
```

```
          .text
_PUTLONG::
          move.l    4(sp),d0
PUTLONG::
          move.l    #10, -(sp)
          move.l    d0, -(sp)
          bsr.s     _ULD
          addq.l    #8, sp
          tst.l     d0
          beq.s     .1
          movem.l   d0-d1,-(sp)
          bsr       PUTLONG
          movem.l   (sp)+, d0-d1
.1:
          addi.w    #$30, d1
          move.w    d1,-(sp)
          move.w    #CCONOUT,-(sp)
          trap      #1
          addq.l    #4,sp
          rts



;*** Unsigned Long to ASCII
;
; e  long value
;    long pointer to string buffer
;
; x  ASCIIZ string in buffer

          .text
_ULTOA::
          move.l    4(sp),d0
          movea.l   8(sp), a0
ULTOA::
          move.l    #10, -(sp)
          move.l    d0, -(sp)
          bsr.s     _ULD
          addq.l    #8, sp
          tst.l     d0
          beq.s     .1
          movem.l   d0-d1,-(sp)
          bsr       ULTOA
          movem.l   (sp)+, d0-d1
.1:
          addi.w    #$30, d1
          move.b    d1, (a0)+
          clr.b     (a0)
          rts


;*** Unsigned long division
;
```

```
; e  long dividend
;    long divisor
;
; x  D0.L = quoitent
;    D1.L = remainder

          .text
_ULD::
          move.l    4(sp), d0           ;dividend
          move.l    8(sp), d2           ;divisor
          move.w    d3, -(sp)
          move.w    #31, d3             ;bit count
          clr.l     d1                  ;remainder
.1:
          add.l     d0, d0
          addx.l    d1, d1
          sub.l     d2, d1
          bcc.s     .2
          add.l     d2, d1
          bra.s     .3
.2:
          addq.l    #1, d0
.3:
          dbf       d3, .1
          move.w    (sp)+, d3
          rts


;*** DOS Function Call (CP/M-68k)
;
; e  word function number
;    long parameter
;
; x  D0.L = returned value

          .text
___BDOS::
          move.w    4(sp),d0       ; get function number
          move.l    8(sp),d1       ; get parameter
          trap      #2             ; call DOS
          rts


          .globl    _gemdos

;*** GEM Dos Function Call
;
; e  word function number
;    word / long parameters
;
; x  D0 = return value if any
;
;    See GEM Dos notes
```

```
          .text
_gemdos:
          move.l    (sp)+, GEMRET
          trap      #1
          move.l    GEMRET, -(sp)
          rts

          .bss
GEMRET:   .ds.l     1



          .globl    _xbios

;*** XBIOS Function Call
;
; e  word function number
;    word / long parameters
;
; x  D0 = return value if any
;
;    See XBIOS notes

          .text
_xbios:
          move.l    (sp)+, XBRET
          trap      #14
          move.l    XBRET, -(sp)
          rts

          .bss
XBRET:    .ds.l     1



@\Rogue\Monster\
else
  echo "shar: Will not over write ACHSLIB.S"
fi
if `test ! -s LIST.C`
then
echo "x - LIST.C"
cat > LIST.C << '@\Rogue\Monster\'
/*
     line.c

     880821   ACH
*/

#include "runtime.h"
```

```
void InsBefore( list, curr, new )
     cell* list;
     cell* curr;
     cell* new;
{
     cell* prev;

     if( !new ) return;
     ++list->x;
     if( curr ){
          prev = curr->prev;
          curr->prev = new;
     } else {
          prev = list->prev;
          list->prev = new;
     }
     new->prev = prev;
     new->next = curr;
     if( prev )
          prev->next = new;
     else
          list->next = new;
}


void InsAfter( list, curr, new )
     cell* list;
     cell* curr;
     cell* new;
{
     cell* next;

     if( !new ) return;
     ++list->x;
     if( curr ){
          next = curr->next;
          curr->next = new;
     } else {
          next = list->next;
          list->next = new;
     }
     new->prev = curr;
     new->next = next;
     if( next )
          next->prev = new;
     else
          list->prev = new;
}


void DelCurr( list, curr )
     cell* list;
     cell* curr;
```

```
{
     cell* next;
     cell* prev;

     if( !curr ) return;
     --list->x;
     prev = curr->prev;
     next = curr->next;
     if( prev )
          prev->next = next;
     else
          list->next = next;
     if( next )
          next->prev = prev;
     else
          list->prev = prev;
}


@\Rogue\Monster\
else
  echo "shar: Will not over write LIST.C"
fi
if `test ! -s MAKEFILE`
then
echo "x - MAKEFILE"
cat > MAKEFILE << '@\Rogue\Monster\'
mtest.prg: achapp.o mtest.o list.o memory.o achslib.o
     aln -c d:mtest.lnk

achapp.o: achapp.s
    mac -u -s -v -oachapp.o achapp.s

achslib.o: achslib.s
     mac  -u -s -v -oachslib.o achslib.s

list.o: list.c
     cc -c list.c

mtest.o: mtest.c
     cc -c mtest.c

memory.o: memory.c
     cc -c memory.c

@\Rogue\Monster\
else
  echo "shar: Will not over write MAKEFILE"
fi
if `test ! -s MEMORY.C`
then
echo "x - MEMORY.C"
cat > MEMORY.C << '@\Rogue\Monster\'
```

```
/*
     memory.c      Anthony's Memory Management Routines

     880824   ACH
*/

#include "runtime.h"

/* BLOCK_SIZE is the size of heaps to be requested from the system. This value
   should be even and in the range 2 <= BLOCK_SIZE <= 2^n-1 -2. */

#define  BLOCK_SIZE    65534
#define BLOCK_SIZE_L 65534L

cell     Avail = { 0, NIL, NIL };
unsigned CellsUsed = 0;
unsigned TotalHeaps = 0;


static bool Mavail( curr )
/*
     Return TRUE if cell is available.
*/
     cell* curr;
{
     if( !curr ) return( FALSE );
     return( !(curr->x & 1) );
}


static cell* Madjcent( curr )
/*
     Return address of right adjcent cell.
*/
     cell* curr;
{
     if( curr ) return( (cell*) ((char*) curr + (curr->x & ~1)) );
     return( NIL );
}


static cell* Mcore()
/*
     Allocate another heap block from the system and add to free list.
     Return pointer to new block or NIL if out of system memory.
*/
{
     cell*    p;

     p = (cell*) SysAlloc( BLOCK_SIZE_L );
     if( p ){
          p->x = BLOCK_SIZE | 1;
          ++CellsUsed;
```

```
          Mdispose( (cell*) ((char*) p + sizeof( unsigned )) );
          ++TotalHeaps;
     }

     return( p );
}


static bool Mrelease( curr )
/*
     Return TRUE is a heap block was released.
*/
     cell* curr;
{
     if( TotalHeaps && curr && curr->x == BLOCK_SIZE ){
          DelCurr( &Avail, curr );
          SysFree( curr );
          --TotalHeaps;
          return( TRUE );
     }

     return( FALSE );
}


bool Mfini()
/*
     We free up all that is left of the heaps. We assume that the programmer
     did all the necessary clean-up so that all the cells merged into heaps.
     Return TRUE if all heaps and cells where released.
*/
{
     cell* p;

     for( p = Avail.next; p; p = p->next )
          Mrelease( p );

     return( !(CellsUsed && TotalHeaps) );
}


void* Mnew( request )
/*
     Return pointer to requested memory. If NIL returned then unsuccessful
     memory request. The range for requests is 1 to BLOCK_SIZE.
*/
     unsigned request;
{
     cell*         p;
     unsigned amount;
     unsigned left;

     /* You don't get anything for nothing. */
```

```
    if( !request ) return( NIL );

    /* Word align our request, so we always ask for even sizes. */
    amount = (request + sizeof( unsigned ) +1) & ~1;

    /* Check for overflow. */
    if( amount <= request || amount > BLOCK_SIZE ) return( NIL );

    /* Min request possible since we have to be able to store pointers
       back into the block when it is later freed. */
    if( amount < sizeof( cell ) ) amount = sizeof( cell );

    /* assert: sizeof( cell ) <= amount <= BLOCK_SIZE */

    /* Traverse free list for first fit. */
    for( p = Avail.next; p; p = p->next ){
        if( Mrelease( p ) ) continue;

        /* assert: sizeof( cell ) <= p->x <= BLOCK_SIZE */
        if( p->x >= amount ){
            allocate:
            left = p->x - amount;
            if( left < sizeof( cell ) )
                DelCurr( &Avail, p );
            else {
                /* Split block and return upper half. */
                p->x = left;
                p = Madjcent( p );
                p->x = amount;
            }
            /* Mark block as used. */
            p->x |= 1;
            ++CellsUsed;

            return( (void*) ((char*) p + sizeof( unsigned )) );
        }
    }

    p = Mcore();
    if( p ){
        /* assert: p->x = BLOCK_SIZE >= amount */
        goto allocate;
    }

    return( NIL );
}


void Mdispose( curr )
/*
    Free cell, return to free list and try and merge with neighbors.
*/
    cell* curr;
```

```
{
     cell*    p;
     cell*    q;

     if( !curr ) return;
     curr = (cell*) ((char*) curr - sizeof( unsigned ));
     /* Mark block as available. */
     curr->x &= ~1;
     --CellsUsed;

     for( p = Avail.next; p < curr && p; p = p->next );
     InsBefore( &Avail, p, curr );

     if( Madjcent( curr ) == p && curr->x != BLOCK_SIZE ){
          /* Merge right neighbor. */
          curr->x += p->x;
          DelCurr( &Avail, p );
     }

     q = curr->prev;
     if( Madjcent( q ) == curr && q->x != BLOCK_SIZE ){
          /* Merge left neighbor. */
          q->x += curr->x;
          DelCurr( &Avail, curr );
     }
}
@\Rogue\Monster\
else
  echo "shar: Will not over write MEMORY.C"
fi
if `test ! -s MTEST.C`
then
echo "x - MTEST.C"
cat > MTEST.C << '@\Rogue\Monster\'
#include <stdio.h>
#include "runtime.h"

void main( argc, argv )
     int argc;
     char** argv;
{
     char** p;
     char* a;
     char* b;
     char* c;
     char* d;
     int i;

     printf( "Check upper limit (max should be 65532)\n" );
     a = Mnew( (unsigned) 65531 );
     b = Mnew( (unsigned) 65532 );
     c = Mnew( (unsigned) 65533 );
     d = Mnew( (unsigned) 65534 );
```

```
     printf( "a= %lx, b= %lx, c= %lx, d= %lx.  C and D should be zero.\n",
               a,b,c,d );
     Mdispose( a );
     Mdispose( b );


     printf( "Copy Argv Test\n\nAllocating  p-array\n" );
     p = (char*) Mnew( sizeof( char* ) * argc );
     printf( "p =  %lx\n", p );

     for( i = 0; i < argc; ++i ){
          printf( "Allocating space for '%s'\n", argv[ i ] );
          p[ i ] = (char*) Mnew( strlen( argv[i] ) +1 );
          strcpy( p[ i ], argv[ i ] );
          printf( "Our copy p[%d] = %lx  '%s'\n", i, p[ i ], p[ i ] );
     }

     for( i = 0; i < argc; ++i ){
          printf( "Releasing p[%d]\n", i );
          Mdispose( p[i] );
     }
     printf( "Releasing p-array\n" );
     Mdispose( p );

     Mfini();
}

@\Rogue\Monster\
else
  echo "shar: Will not over write MTEST.C"
fi
if `test ! -s MTEST.LNK`
then
echo "x - MTEST.LNK"
cat > MTEST.LNK << '@\Rogue\Monster\'
-s -v -u
-o mtest.prg
achapp.o
mtest.o
list.o
memory.o
achslib.o
gemlib
@\Rogue\Monster\
else
  echo "shar: Will not over write MTEST.LNK"
fi
if `test ! -s README`
then
echo "x - README"
cat > README << '@\Rogue\Monster\'
readme   88.09.20
```

files in archive:

```
achacc.s      Anthony's MADMAC Assembler Library for Acylon C
achapp.s
achslib.h
achslib.s

list.c            Doubly Link List Routines  - InsBefore, InsAfter, DelCurr
memory.c      Memory Management Routines - Mnew, Mdispose, Mfini
runtime.h     Data types and funtion prototypes.

makefile      Make file for mtest.prg
mtest.c           Memory management test program.
mtest.lnk     Aln link file

readme            This file.
```

I've provided my assembler library for those who still use Acylon C (like me). It has been modified since the last release. Basically adopting standard C library names and parameter conventions.

List.c provides generic doubly link list routines that are used by Memory.c to mantain the free list. Runtime.h contains all the data types used and function prototypes.

Reference for that helped in the development of Memory.c:

1) Knuth vol 1, pg 435-451
2) Kernighan & Ritchie "The C Progamming Language" pg 173-177

Memory.c uses unsigned int which limits memory requests to Mnew in the range 1-65534. This is because Acylon C doesn't support unsigned long. Also the application for which this was written did not require large memory blocks. The source could be modified to fix this limitation if compiled under MWC or GNU CC.

Notes:

1) All source used tabstops = 4
2) Development site was Colour Mega ST 2
3) Development software: BDT Micro C-Shell, BDT Make, LV, Acylon C, MadMac Assembler, Aln Linker

Any inquires send mail to:

achowe@watmsg.waterloo.edu

Disclaimer: The software provided has been written out of need and a desire to learn. Everything has been tested, yet things can go wrong (ie. typos). I refuse to take any responsablity for damage that could have resulted from the use of this code. I have enough trouble finding a decent pizza at 1 am.

PLEASE PLEASE test code independantly till you get that warm feeling like
when you p__s in a pool :)

Anthony C. Howe

```
@\Rogue\Monster\
else
  echo "shar: Will not over write README"
fi
if `test ! -s RUNTIME.H`
then
echo "x - RUNTIME.H"
cat > RUNTIME.H << '@\Rogue\Monster\'
/*
     runtime.h

     880821    ACH
*/

#undef ANSI
#define  ATARI
#define ACYLON
#define  SCREEN_WIDTH  80
#define TAB_WIDTH      4


#ifndef ANSI
typedef int   void;
#endif

#ifdef ATARI
#include <gembind.h>
#include <gemdefs.h>
#include <osbind.h>
#include <tosdefs.h>
#include <vdibind.h>
#include "achslib.h"

#define  putch(c)  (Cconout(c))
#define Key()      (Cconin())
#define SysAlloc(x)    (Malloc(x))
#define SysFree(x) (Mfree(x))
#define READ       0
#define WRITE      1
#define READ_WRITE     2

typedef struct dta
{
     char reserved[21];
     char attr;
     short    time;
```

```
     short    date;
     long size;
     char name[14];
}
     dta;

#else
#define  READ      "r"
#define WRITE      "w"
#define READ_WRITE "rw"
#endif

#undef   FALSE
#define  FALSE    0
#undef   TRUE
#define  TRUE (!FALSE)
#undef   NULL
#define  NULL 0
#undef   NIL
#define  NIL      0L


typedef int        bool;


typedef struct cell
{
     unsigned      x;
     struct cell*  prev;
     struct cell*  next;
}
     cell;


#ifdef ANSI
/* list.c */
extern void InsBefore( cell* list, cell* curr, cell* new );
extern void InsAfter( cell* list, cell* curr, cell*, new );
extern void DelCurr( cell* list, cell* curr );

/*   memory.c */
extern bool Mfini();
extern void* Mnew( unsigned request );
extern void Mdispose( cell* curr );

#else
/* list.c */
extern void InsBefore();
extern void InsAfter();
extern void DelCurr();
extern cell* Lalloc();
extern void Lfree();
extern void Lfini();
```

```
/*   memory.c */
extern bool Mfini();
extern void* Mnew();
extern void Mdispose();
#endif


@\Rogue\Monster\
else
  echo "shar: Will not over write RUNTIME.H"
fi
# to concatenate archives, remove anything after this line
exit 0


From 2581901074900923209
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,edd2af90e7ac3bb8,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-11-30 21:15:53 PST
Path: utzoo!attcan!uunet!wyse!vsi1!ames!pasteur!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i076: sttar -- Unix-style "tar" utility
Keywords: shar
Message-ID: <5550@saturn.ucsc.edu>
Date: 21 Nov 88 10:26:20 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1535
Approved: koreth@ssyx.ucsc.edu

Submitted-by: cs.buffalo.edu!sigmast!dgy (Dave Yearke)
Posting-number: Volume 1, Issue 76
Archive-name: sttar

[This arrived in the form of three shar files inside an arc file; I unpacked
 the arc file and sharred the shar files.  The result is that you'll have to
 unshar four times to get everything out.  Let me know if this causes any
 problems. -sg]

This is an implementation of "tar" for the ST.  Questions and comments
should be directed to Carl Barron at sigmast!brass!carl_barron.

#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      ctar.shr
#      dtar.shr
#      stftw.shr
#
sed 's/^X//' << 'SHAR_EOF' > ctar.shr &&
```

```
X#! /bin/sh
X# This is a shell archive, meaning:
X# 1. Remove everything above the #! /bin/sh line.
X# 2. Save the resulting text in a file.
X# 3. Execute the file with /bin/sh (not csh) to create the files:
X#   createta.c
X#   ctar.c
X#   detar.h
X#   makefile
X#   todo.c
X#   tounix.c
X# This archive created: Fri Aug 19 03:21:53 1988
Xexport PATH; PATH=/bin:$PATH
Xif test -f 'createta.c'
Xthen
X    echo shar: will not over-write existing file "'createta.c'"
Xelse
Xcat << \SHAR_EOF > 'createta.c'
X#include <stdio.h>
X#include "ndir.h"
X#include "ftw.h"
X#include "detar.h"
Xextern FILE *ftar;
X
Xextern char *tounix(); /* char * */
Xextern void tooct(),calc_chksum();
Xstatic union record tar_rec;
Xstatic struct stat *stbuf;
X
Xvoid bzero(p,n)char *p;int n;
X{
X    while(n--)
X         *p++='\0';
X}
X
Xstatic void dump_header(name,x)char *name;int x;
X{
X         bzero(tar_rec.charptr,RECORDSIZE);
X         strncpy(tar_rec.header.name,tounix(name),NAMSIZ);
X         tooct(0644L,8,tar_rec.header.mode);
X         tooct(0L,8,tar_rec.header.uid);
X         tooct(0L,8,tar_rec.header.gid);
X         tooct(stbuf->st_size,12,tar_rec.header.size);
X         tooct(stbuf->st_mtime,12,tar_rec.header.mtime);
X         strncpy(tar_rec.header.magic,TMAGIC,8);
X         strncpy(tar_rec.header.chksum,CHKBLANKS,8);
X         tar_rec.header.linkflag=x+'0';
X         calc_chksum(&tar_rec);
X         fwrite(tar_rec.charptr,1,RECORDSIZE,ftar);
X}
X
Xstatic void dump_file(name)char *name;
X{
```

```
X    FILE *fin;
X    int n;
X
X    fin=fopen(name,"rb");
X
X    do
X    {
X         bzero(tar_rec.charptr,RECORDSIZE);
X         n=fread(tar_rec.charptr,1,RECORDSIZE,fin);
X         fwrite(tar_rec.charptr,1,RECORDSIZE,ftar);
X    }while(n==RECORDSIZE);
X    fclose(fin);
X}
X
Xint create_tar(name,sbuf,flag)char *name;struct stat *sbuf;int flag;
X{
X    stbuf=sbuf;
X    switch(flag)
X    {
X    case FTW_NS:
X         fprintf(stderr,"ctar:can't stat %s\n",name);
X         return -1;
X    case FTW_DNR:
X         fprintf(stderr,"ctar:can't open directory %s\n");
X         return -2;
X    case FTW_F:
X         dump_header(name,0);
X         dump_file(name);
X         return 0;
X    case FTW_D:
X         strcat(name,"\\");
X         dump_header(name,5);
X         return 0;
X    }
X}
X
XSHAR_EOF
Xchmod +x 'createta.c'
Xfi # end of overwriting check
Xif test -f 'ctar.c'
Xthen
X    echo shar: will not over-write existing file "'ctar.c'"
Xelse
Xcat << \SHAR_EOF > 'ctar.c'
X#include <stdio.h>
X
XFILE *ftar;
X
Xextern int create_tar();
X
Xmain(ac,av)int ac;char *av[];
X{
X    ftar=fopen(av[1],"wb");
```

```
X    ac -= 2;
X    av += 2;
X
X    while(ac--)
X    {
X         if(ftw(*av++,create_tar,10))break;
X    }
X    fclose(ftar);
X}
XSHAR_EOF
Xchmod +x 'ctar.c'
Xfi # end of overwriting check
Xif test -f 'detar.h'
Xthen
X    echo shar: will not over-write existing file "'detar.h'"
Xelse
Xcat << \SHAR_EOF > 'detar.h'
X/*
X *  MSDOS TAR Extractor
X */
X
X#define  RECORDSIZE    512
X#define  NAMSIZ        100
X#define  TUNMLEN       32
X#define  TGNMLEN       32
X
X/*
X *  Header block on tape.
X *
X *  no byte swapping
X */
X
Xunion record {
X    char charptr[RECORDSIZE];
X    struct   {
X         char name[NAMSIZ];
X         char mode[8];
X         char uid[8];
X         char gid[8];
X         char size[12];
X         char mtime[12];
X         char chksum[8];
X         char linkflag;
X         char linkname[NAMSIZ];
X         char magic[8];
X         char uname[TUNMLEN];
X         char gname[TGNMLEN];
X         char devmajor[8];
X         char devminor[8];
X    } header;
X};
X
X#define  CHKBLANKS"        "    /* Checksum: 8 blanks, no null */
```

```
X#define TMAGIC        "ustar  " /* Majic: 7 bytes and a null */
X
X/* The linkflag defines the type of file */
X
X#define LF_OLDNORMAL  '\0'      /* Normal disk file, Unix compat */
X#define LF_NORMAL '0'       /* Normal disk file */
X#define LF_LINK       '1'       /* Link to previously dumped file */
X#define LF_SYMLINK    '2'       /* Symbolic link */
X#define LF_CHR        '3'       /* Character special file */
X#define LF_BLK        '4'       /* Block special file */
X#define LF_DIR        '5'       /* Directory */
X#define LF_FIFO       '6'       /* FIFO special file */
X#define LF_CONTIG '7'       /* Contiguous file */
X
X/*
X * Unix Stat Header (K&R)
X *
X */
X
Xstruct  stat  {
X    int  st_dev;
X    int  st_ino;
X    int  st_mode;
X    int  st_nlink;
X    int  st_uid;
X    int  st_gid;
X    int  st_rdev;
X    long st_size;
X    long st_atime;
X    long st_mtime;
X    long st_ctime;
X};
X
X#define S_IFMT        0160000
X#define S_IFDIR       0040000
X#define S_IFCHR       0020000
X#define S_IFBLK       0060000
X#define S_IFREG       0100000
X#define S_ISUID       0004000
X#define S_ISGID       0002000
X#define S_ISVTX       0001000
X#define S_IREAD       0000400
X#define S_IWRITE  0000200
X#define S_IEXEC       0000100
XSHAR_EOF
Xchmod +x 'detar.h'
Xfi # end of overwriting check
Xif test -f 'makefile'
Xthen
X    echo shar: will not over-write existing file "'makefile'"
Xelse
Xcat << \SHAR_EOF > 'makefile'
XCFLAGS=-A
```

```
XOBJ=ctar.o createta.o tounix.o todo.o
Xctar.prg: $(OBJ)
X    cc $(OBJ) -ldir
XSHAR_EOF
Xchmod +x 'makefile'
Xfi # end of overwriting check
Xif test -f 'todo.c'
Xthen
X    echo shar: will not over-write existing file "'todo.c'"
Xelse
Xcat << \SHAR_EOF > 'todo.c'
X#include "detar.h"
Xvoid tooct();
X
Xvoid bcopy(src,dest,n)char *src,*dest;int n;
X{
X    while(n--)
X         *dest++ = *src++;
X}
X
Xvoid
Xcalc_chksum(header)
X    register union record *header;
X{
X    register int  i, sum;
X    register char *p;
X
X    bcopy(CHKBLANKS, header->header.chksum, sizeof(header->header.chksum));
X
X    sum = 0;
X    p = header->charptr;
X    for (i = sizeof(*header); --i >= 0; ) {
X         /*
X          * We can't use unsigned char here because of old compilers,
X          * e.g. V7.
X          */
X         sum += 0xFF & *p++;
X    }
X
X    /*
X     * Fill in the checksum field.  It's formatted differently
X     * from the other fields:  it has [6] digits, a null, then a
X     * space -- rather than digits, a space, then a null.
X     * We use tooct then write the null in over tooct's space.
X     * The final space is already there, from checksumming, and
X     * tooct doesn't modify it.
X     *
X     * This is a fast way to do:
X     * (void) sprintf(header->header.chksum, "%6o", sum);
X     */
X    tooct((long) sum,  8,  header->header.chksum);
X    header->header.chksum[6] = '\0'; /* Zap the space */
X}
```

```
X
Xvoid
Xtooct(value, digs, where)
X    register long value;
X    register int  digs;
X    register char *where;
X{
X
X    --digs;                /* Trailing null slot is left alone */
X    where[--digs] = ' ';        /* Put in the space, though */
X
X    /* Produce the digits -- at least one */
X    do {
X         where[--digs] = '0' + (char)(value & 7); /* one octal digit */
X         value >>= 3;
X    } while (digs > 0 && value != 0);
X
X    /* Leading spaces, if necessary */
X    while (digs > 0)
X         where[--digs] = ' ';
X
X}
XSHAR_EOF
Xchmod +x 'todo.c'
Xfi # end of overwriting check
Xif test -f 'tounix.c'
Xthen
X    echo shar: will not over-write existing file "'tounix.c'"
Xelse
Xcat << \SHAR_EOF > 'tounix.c'
Xchar *tounix(name)register char *name;
X{
X    static char unix[100];
X    register char *p;
X
X    bzero(unix,100);
X    for(p=unix;*name;name++,p++)
X         switch(*name)
X         {
X         case '_':
X              *p='B';
X              break;
X         case '\\':
X              *p='/';
X              break;
X         default:
X              *p = *name;
X              break;
X         }
X    return unix;
X}
XSHAR_EOF
Xchmod +x 'tounix.c'
```

```
Xfi # end of overwriting check
X#   End of shell archive
Xexit 0
SHAR_EOF
chmod 0600 ctar.shr || echo "restore of ctar.shr fails"
sed 's/^X//' << 'SHAR_EOF' > dtar.shr &&
X#! /bin/sh
X# This is a shell archive, meaning:
X# 1. Remove everything above the #! /bin/sh line.
X# 2. Save the resulting text in a file.
X# 3. Execute the file with /bin/sh (not csh) to create the files:
X#   detar.man
X#   makefile
X#   detar.c
X#   detar.h
X#   dodetar.c
X#   misc.c
X#   togem.c
X# This archive created: Mon Feb  8 14:37:02 1988
Xexport PATH; PATH=/bin:$PATH
Xif test -f 'detar.man'
Xthen
X    echo shar: will not over-write existing file "'detar.man'"
Xelse
Xcat << \SHAR_EOF > 'detar.man'
X
X    NAME - detar extract all files or list files in a tar file
X
X    USAGE - detar [d] tarfile
X
X    if d is specified a verbose listing of the files in the tar file
X    is given and no extraction occurs. Otherwise ALL the files in the
X    tar file are extracted into directories as per the tar file.
X
XSHAR_EOF
Xchmod +x 'detar.man'
Xfi # end of overwriting check
Xif test -f 'makefile'
Xthen
X    echo shar: will not over-write existing file "'makefile'"
Xelse
Xcat << \SHAR_EOF > 'makefile'
XCFLAGS=-A
XOFLAGS=
XLIB=
XPROG=detar.ttp
XOBJ=\
Xdetar.o\
Xdodetar.o misc.o togem.o
X$(PROG): $(OBJ)
X    cc -o $(PROG) $(OFLAGS) $(OBJ) $(LIB)
Xdetar.o: detar.h
Xdodetar.o: detar.h
```

```
XSHAR_EOF
Xchmod +x 'makefile'
Xfi # end of overwriting check
Xif test -f 'detar.c'
Xthen
X    echo shar: will not over-write existing file "'detar.c'"
Xelse
Xcat << \SHAR_EOF > 'detar.c'
X#include <stdio.h>
X#include "detar.h"
Xchar **extract;
Xint extsize;
XFILE *ftar;
Xextern int all(),some(),strcmp();
X
Xmain(ac,av)int ac;char *av[];
X{
X    if(ac<2)
X    {
X         fprintf(stderr,"detar: tarfile [file(s)]\n");
X         exit(1);
X    }
X    if((ftar=fopen(av[1],"rb"))==NULL)
X    {
X         fprintf(stderr,"detar:Can't open tarfile %s\n",av[1]);
X         exit(1);
X    }
X    if(ac==2)
X         do_detar(all);
X    else
X    {
X         ac -= 2;
X         av += 2;
X         extsize=ac;
X         extract=av;
X         qsort(extract,ac,sizeof(char *),strcmp);
X         do_detar(some);
X    }
X    fclose(ftar);
X}
X
Xint all(what)char *what;
X{
X    return 1;
X}
X
Xint some(what)char *what;
X{
X    int c,i;
X
X    for(i=0;i<extsize;i++)
X    {
X         if((c=strcmp(what,extract[i]))<0)
```

```
X             return 0;
X         if(c==0) return 1;
X    }
X    return 0;
X}
XSHAR_EOF
Xchmod +x 'detar.c'
Xfi # end of overwriting check
Xif test -f 'detar.h'
Xthen
X    echo shar: will not over-write existing file "'detar.h'"
Xelse
Xcat << \SHAR_EOF > 'detar.h'
X/*
X *  MSDOS TAR Extractor
X */
X
X#define  RECORDSIZE    512
X#define  NAMSIZ        100
X#define  TUNMLEN       32
X#define  TGNMLEN       32
X
X/*
X *  Header block on tape.
X *
X *  no byte swapping
X */
X
Xunion record {
X    char charptr[RECORDSIZE];
X    struct   {
X         char name[NAMSIZ];
X         char mode[8];
X         char uid[8];
X         char gid[8];
X         char size[12];
X         char mtime[12];
X         char chksum[8];
X         char linkflag;
X         char linkname[NAMSIZ];
X         char magic[8];
X         char uname[TUNMLEN];
X         char gname[TGNMLEN];
X         char devmajor[8];
X         char devminor[8];
X    } header;
X};
X
X#define  CHKBLANKS"        "    /* Checksum: 8 blanks, no null */
X#define  TMAGIC        "ustar  "/* Majic: 7 bytes and a null */
X
X/* The linkflag defines the type of file */
X
```

```
X#define LF_OLDNORMAL  '\0'      /* Normal disk file, Unix compat */
X#define LF_NORMAL '0'       /* Normal disk file */
X#define LF_LINK       '1'       /* Link to previously dumped file */
X#define LF_SYMLINK    '2'       /* Symbolic link */
X#define LF_CHR        '3'       /* Character special file */
X#define LF_BLK        '4'       /* Block special file */
X#define LF_DIR        '5'       /* Directory */
X#define LF_FIFO       '6'       /* FIFO special file */
X#define LF_CONTIG '7'       /* Contiguous file */
X
X/*
X * Unix Stat Header (K&R)
X *
X */
X
Xstruct  stat  {
X    int  st_dev;
X    int  st_ino;
X    int  st_mode;
X    int  st_nlink;
X    int  st_uid;
X    int  st_gid;
X    int  st_rdev;
X    long st_size;
X    long st_atime;
X    long st_mtime;
X    long st_ctime;
X};
X
X#define S_IFMT        0160000
X#define S_IFDIR       0040000
X#define S_IFCHR       0020000
X#define S_IFBLK       0060000
X#define S_IFREG       0100000
X#define S_ISUID       0004000
X#define S_ISGID       0002000
X#define S_ISVTX       0001000
X#define S_IREAD       0000400
X#define S_IWRITE  0000200
X#define S_IEXEC       0000100
XSHAR_EOF
Xchmod +x 'detar.h'
Xfi # end of overwriting check
Xif test -f 'dodetar.c'
Xthen
X    echo shar: will not over-write existing file "'dodetar.c'"
Xelse
Xcat << \SHAR_EOF > 'dodetar.c'
X#include <stdio.h>
X#include "detar.h"
Xstatic union record tar_rec;
Xstatic struct s_q
X{
```

```
X    char name[NAMSIZ];
X    char link[NAMSIZ];
X    int mode,linkflag;
X    long size;
X}my_header;
X
Xstatic struct link_q
X{
X    struct link_q *prev;
X    char name[NAMSIZ];
X    char link[NAMSIZ];
X}*link_list,*new_link,*todo_link;
X
Xextern long from_oct();
Xextern int calc_chksum();
Xextern FILE *ftar;
Xextern char *to_gem();
Xstatic int read_header()
X{
X    int chk;
X    do
X    {
X         fread(tar_rec.charptr,1,RECORDSIZE,ftar);
X         if(feof(ftar))return -1;
X         chk=calc_chksum(&tar_rec);
X    }while(chk==2);
X    return chk;
X}
X
Xdo_detar(test)int (*test)();
X{
X    int chk;
X    link_list=NULL;
X    while((chk=read_header())>=0)
X    {
X         if(chk==0)
X         {
X              fprintf(stderr,
X              "Checksum error on %s aborting",tar_rec.header.name);
X              exit(2);
X         }
X         if(tar_rec.header.name[strlen(tar_rec.header.name)-1]=='/')
X         {
X              tar_rec.header.name[strlen(tar_rec.header.name)-1]='\0';
X              my_header.linkflag=LF_DIR;
X         }
X         else
X              my_header.linkflag=tar_rec.header.linkflag;
X         strcpy(my_header.name,to_gem(tar_rec.header.name));
X         if(!(*test)(my_header.name))
X              continue;
X         strcpy(my_header.link,to_gem(tar_rec.header.linkname));
X         my_header.size=from_oct(12,tar_rec.header.size);
```

```
X         switch(my_header.linkflag)
X         {
X         case LF_OLDNORMAL:
X         case LF_NORMAL:
X         case LF_CHR:
X         case LF_BLK:
X         case LF_FIFO:
X         case LF_CONTIG:
X         {
X              FILE *fout;
X              int n;
X              Makedirs(my_header.name);
X              if((fout=fopen(my_header.name,"wb"))==NULL)
X              {
X                   fprintf(stderr,"detar:Can't open %s\n",
X                        my_header.name);
X                   exit(3);
X              }
X              while(my_header.size >0)
X              {
X                   fread(tar_rec.charptr,1,RECORDSIZE,ftar);
X                   n=(my_header.size>RECORDSIZE)?RECORDSIZE:
X                        (int)my_header.size;
X                   fwrite(tar_rec.charptr,1,n,fout);
X                   my_header.size = my_header.size - RECORDSIZE;
X
X              }
X              fclose(fout);
X              break;
X         }
X         case LF_LINK:
X         case LF_SYMLINK:
X              new_link=(struct link_q *)malloc(sizeof(struct link_q));
X              new_link->prev=link_list;
X              strcpy(new_link->name,my_header.name);
X              strcpy(new_link->link,my_header.link);
X              link_list=new_link;
X              break;
X         case LF_DIR:
X              mkdir(my_header.name);
X              break;
X         }
X    }
X    do
X    {
X         todo_link=NULL;
X         while(link_list)
X         {
X              struct stat stbuf;
X
X              new_link=link_list;
X              link_list=new_link->prev;
X              if(stat(new_link->link,&stbuf))
```

```
X             {
X                  new_link->prev=todo_link;
X                  todo_link=new_link;
X             }
X             else
X             {
X                  FILE *fin,*fout;
X                  int n;
X                  Makedirs(new_link->name);
X                  if((fin=fopen(new_link->link,"rb"))==NULL)
X                  {
X                       fprintf(stderr,
X                       "Detar:Cant open %s for copying\n",
X                            new_link->link);
X                       exit(3);
X                  }
X                  if((fout=fopen(new_link->name,"wb"))==NULL)
X                  {
X                       fprintf(stderr,
X                       "Detar:Can't open %s to copy to\n",
X                            new_link->name);
X                       exit(3);
X                  }
X                  for(n=RECORDSIZE;n;)
X                  {
X                       n=fread(tar_rec.charptr,1,RECORDSIZE,
X                            fin);
X                       fwrite(tar_rec.charptr,1,n,fout);
X                  }
X                  fclose(fin);
X                  fclose(fout);
X                  free(new_link);
X             }
X        }
X        link_list=todo_link;
X    }while(todo_link);
X}
XSHAR_EOF
Xchmod +x 'dodetar.c'
Xfi # end of overwriting check
Xif test -f 'misc.c'
Xthen
X    echo shar: will not over-write existing file "'misc.c'"
Xelse
Xcat << \SHAR_EOF > 'misc.c'
X#include <ctype.h>
X#include "detar.h"
X#define isodigit(x) (isdigit(x) && ((x)<'8'))
X
Xvoid bcopy(src,dest,n)char *src,*dest;int n;
X{
X    while(n--)
X         *dest++ = *src++;
```

```
X}
X
Xlong from_oct(digs,where)int digs;char *where;
X{
X    long ans=0;
X
X    while(isspace(*where) && digs)
X         digs--,where++;
X    if(!digs) return 0L;
X
X    while(isodigit(*where) && digs)
X    {
X         ans = ans *8 +(*where++ - '0');
X         digs--;
X    }
X    return ans;
X}
X
Xint calc_chksum(tar_rec) union record *tar_rec;
X{
X    long sum,recsum;
X    int i;
X    char *p;
X
X    p=tar_rec->charptr;
X    recsum=from_oct(8,tar_rec->header.chksum);
X    sum=0L;
X    for(i=sizeof(*tar_rec);--i>=0;)
X         sum = sum + (*p++ & 0xff);
X    for(i=sizeof(tar_rec->header.chksum);--i>=0;)
X         sum = sum - (0xff & tar_rec->header.chksum[i]);
X    sum = sum +' '*sizeof(tar_rec->header.chksum);
X    if(sum==recsum) return 1;
X    if(sum==8*' ') return 2;
X    return 0;
X}
X
Xvoid Makedirs(name)char *name;
X{
X    struct stat stbuf;
X    char *p,*q,*index();
X    int makeit;
X
X    p=name;
X    while(q=index(p,'\\'))
X    {
X         *q='\0';
X         makeit=0;
X         if(stat(name,&stbuf))
X              makeit=1;
X         else if((stbuf.st_mode & S_IFMT) != S_IFDIR)

X              makeit=1;
```

```
X         if(makeit)
X              mkdir(name);
X         *q='\\';
X         p=q+1;
X    }
X}
X
X
XSHAR_EOF
Xchmod +x 'misc.c'
Xfi # end of overwriting check
Xif test -f 'togem.c'
Xthen
X    echo shar: will not over-write existing file "'togem.c'"
Xelse
Xcat << \SHAR_EOF > 'togem.c'
Xextern char *rindex(),*index();
X
Xvoid bzero(p,n)char *p;int n;
X{
X    while(n--)
X         *p++='\0';
X}
X
Xstatic char *do_one(s)char *s;
X{
X    static char work[16];
X    char *p,*q;
X
X    bzero(work,16);
X
X    if(p=rindex(s,'.'))
X    {
X         *p='\0';
X         strncpy(work,s,8);
X         for(q=work;*q;q++)
X              if(*q=='.') *q='_';
X         *p='.';
X         strncat(work,p,4);
X    }else
X         strncpy(work,s,8);
X    return work;
X}
X
Xchar *to_gem(unix)char *unix;
X{
X    static char gem[100];
X    char *p,*q;
X
X    bzero(gem,100);
X    p=unix;
X    while(q=index(p,'/'))
X    {
```

```
X        *q='\0';
X        strcat(gem,do_one(p));
X        strcat(gem,"\\");
X        *q='/';
X        p=q+1;
X    }
X    strcat(gem,do_one(p));
X}
X
X
X#ifdef TEST
Xmain(ac,av)int ac;char *av[];
X{
X    printf("Unix: %s\nGem:  %s\n",av[1],to_gem(av[1]));
X}
X#endif
XSHAR_EOF
Xchmod +x 'togem.c'
Xfi # end of overwriting check
X#   End of shell archive
Xexit 0
SHAR_EOF
chmod 0600 dtar.shr || echo "restore of dtar.shr fails"
sed 's/^X//' << 'SHAR_EOF' > stftw.shr &&
X#! /bin/sh
X# This is a shell archive, meaning:
X# 1. Remove everything above the #! /bin/sh line.
X# 2. Save the resulting text in a file.
X# 3. Execute the file with /bin/sh (not csh) to create the files:
X#   alphasor.c
X#   ftw.c
X#   ftw.h
X#   ftw.man
X#   ndir.c
X#   ndir.h
X#   ndir.man
X#   scandir.c
X#   scandir.man
X# This archive created: Mon Feb  8 14:17:43 1988
Xexport PATH; PATH=/bin:$PATH
Xif test -f 'alphasor.c'
Xthen
X    echo shar: will not over-write existing file "'alphasor.c'"
Xelse
Xcat << \SHAR_EOF > 'alphasor.c'
X/*
X**  ALPHASORT
X**  Trivial sorting predicate for scandir; puts entries in alphabetical order.
X*/
X#include <sys/types.h>
X#include <sys/dir.h>
X#ifdef  RCSID
Xstatic char RCS[] = "$Header: alphasort.c,v 1.1 87/12/29 21:35:59 rsalz Exp $";
```

```
X#endif  /* RCSID */
X
X/* A convenient shorthand. */
Xtypedef struct direct  ENTRY;
X
Xint
Xalphasort(d1, d2)
X    ENTRY    **d1;
X    ENTRY    **d2;
X{
X    return(strcmp(d1[0]->d_name, d2[0]->d_name));
X}
XSHAR_EOF
Xchmod +x 'alphasor.c'
Xfi # end of overwriting check
Xif test -f 'ftw.c'
Xthen
X    echo shar: will not over-write existing file "'ftw.c'"
Xelse
Xcat << \SHAR_EOF > 'ftw.c'
X/*
X**  FTW
X**  Walk a directory hierarchy from a given point, calling a user-supplied
X**  function at each thing we find.  If we go below a specified depth,
X**  recycle file descriptors.
X*/
X#include <stdio.h>
X
X#ifdef MWC
X#include <types.h>
X#include <stat.h>
X#include "ndir.h"
X#include "ftw.h"
X#define SEP '\\'
X#else
X#include <sys/types.h>
X#include <sys/stat.h>
X#include <sys/dir.h>
X#include <ftw.h>
X#define SEP '/'
X#endif
X
X#ifdef  RCSID
Xstatic char RCS[] = "$Header: ftw.c,v 1.1 87/12/29 21:38:52 rsalz Exp $";
X#endif  /* RCSID */
X
X/* Handy shorthand. */
X#define EQ(a, b)  (strcmp((a), (b)) == 0)
X
X/* Linked in later. */
Xextern char      *malloc();
Xextern char      *strcpy();
X
```

```
X
Xint
Xftw(directory, funcptr, depth)
X    char      *directory;
X    int           (*funcptr)();
X    int             depth;
X{
X    register DIR   *Dp;
X    register char  *p;
X    register int    i;
X    struct direct  *E;
X    struct stat         Sb;
X    long       seekpoint;
X    char      *fullpath;
X
X    /* If can't stat, tell the user so. */
X    if (stat(directory, &Sb) < 0)
X    return((*funcptr)(directory, &Sb, FTW_NS));
X
X    /* If it's not a directory, call the user's function. */
X    if ((Sb.st_mode & S_IFMT) != S_IFDIR)
X    /* Saying "FTW_F" here is lying, what if this is a symlink? */
X    return((*funcptr)(directory, &Sb, FTW_F));
X
X    /* Open directory; and if we can't tell the user so. */
X    if ((Dp = opendir(directory)) == NULL)
X    return((*funcptr)(directory, &Sb, FTW_DNR));
X
X    /* See if user wants to go further. */
X    if (i = (*funcptr)(directory, &Sb, FTW_D)) {
X    closedir(Dp);
X    return(i);
X    }
X
X    /* Get ready to hold the full paths. */
X    i = strlen(directory);
X    if ((fullpath = malloc(i + 1 + MAXNAMLEN + 1)) == NULL) {
X    closedir(Dp);
X    return(-1);
X    }
X    (void)strcpy(fullpath, directory);
X    p = &fullpath[i];
X    if (i && p[-1] != SEP)
X    *p++ = SEP;
X
X    /* Read all entries in the directory.. */
X    while (E = readdir(Dp))
X    if (!EQ(E->d_name, ".") && !EQ(E->d_name, "..")) {
X        if (depth <= 1) {
X         /* Going too deep; checkpoint and close this directory. */
X         seekpoint = telldir(Dp);
X         closedir(Dp);
X         Dp = NULL;
```

```
X        }
X
X        /* Process the file. */
X        (void)strcpy(p, E->d_name);
X        if (i = ftw(fullpath, funcptr, depth - 1)) {
X         /* User's finished; clean up. */
X         free(fullpath);
X         if (Dp)
X             closedir(Dp);
X         return(i);
X        }
X
X        /* Reopen the directory if necessary. */
X        if (Dp == NULL) {
X         if ((Dp = opendir(directory)) == NULL) {
X             /* WTF? */
X             free(fullpath);
X             return(-1);
X         }
X         seekdir(Dp, seekpoint);
X        }
X    }
X
X    /* Clean up. */
X    free(fullpath);
X    closedir(Dp);
X    return(0);
X}
XSHAR_EOF
Xchmod +x 'ftw.c'
Xfi # end of overwriting check
Xif test -f 'ftw.h'
Xthen
X    echo shar: will not over-write existing file "'ftw.h'"
Xelse
Xcat << \SHAR_EOF > 'ftw.h'
X/*
X**  <FTW.H>
X**  Header values for the third parameter to the user-supplied function
X**  for ftw().
X**
X**  $Header: ftw.h,v 1.1 87/12/29 21:34:34 rsalz Exp $
X*/
X
X#define FTW_NS        100  /* Something stat(2) failed on
          */
X#define FTW_DNR       200  /* Something opendir(3) failed on
     */
X#define FTW_F         300  /* A normal file            */
X#define FTW_D         400  /* A directory
     */
XSHAR_EOF
Xchmod +x 'ftw.h'
```

```
Xfi # end of overwriting check
Xif test -f 'ftw.man'
Xthen
X    echo shar: will not over-write existing file "'ftw.man'"
Xelse
Xcat << \SHAR_EOF > 'ftw.man'
X    NAME  ftw - walk a file tree
X
X    SYNOPSIS
X
X#include <ftw.h>
X
Xint
Xftw(directory, funcptr, depth)
X    char *directory;
X    int (*funcptr)();
X    int depth;
X
X#include <stat.h>
X
Xint
Xfuncptr(item, sb, flag)
X    char *item;
X    struct stat *sb;
X    int flag;
X
X
X    DESCRIPTION
X
X    Ftw walks through the directory tree starting from the indicated path .
XFor every entry it finds in the tree, it calls the user-supplied funcptr with
Xthe calling sequence given in the synopsis above.  The first argument is the
Xfull pathname of the entry (rooted from the directory parameter given to ftw );
Xthe second argument is a pointer to the stat (2) structure for the entry;
Xand the third argument is one of the #define's in the header file.
XThis value will be one of the following:
X
X    FTW_F     Item is a normal file
X    FTW_D     Item is a directory
X    FTW_NS    The stat failed on the item
X    FTW_DNR   Item is a directory which can't be read
X
X
XNote, however, that FTW_F is a misnomer; anything other than directories
Xare (e.g., symbolic links) get the FTW_F tag.
X
X    Ftw recursively calls itself when it encounters a directory. To avoid
Xusing up all a program's file descriptors, the depth argument specifies the
Xnumber of simultaneous open directories to maintain.  When the depth is
Xexceeded, the routine will become noticeably slower because directories are
Xclosed in ``most-recently-used'' order.
X
X    To stop the tree walk, the user-supplied function should return a
```

```
Xnon-zero value; this value will become the return value of ftw .  Otherwise,
Xftw will continue until it has scanned the entire tree, in which case it will
Xreturn zero, or until it hits an error such as a malloc (3) failure, in which
Xcase it will return -1.
X
X    Because ftw uses dynamic data structures, the only safe way to exit
Xout of a tree walk is to return a non-zero value.  To handle interrupts, for
Xexample, mark that the interrupt occured and return a non-zero value, don't
Xuse longjmp (3) unless the program is going to terminate.
X
X    SEE ALSO
X
X    stat(2)
XSHAR_EOF
Xchmod +x 'ftw.man'
Xfi # end of overwriting check
Xif test -f 'ndir.c'
Xthen
X    echo shar: will not over-write existing file "'ndir.c'"
Xelse
Xcat << \SHAR_EOF > 'ndir.c'
X#define DEF_NDIR
X#include <stat.h>
X#include <osbind.h>
X#include "ndir.h"
Xextern char *malloc();
X#define MALLOC(x) (x *)malloc(sizeof(x))
X
XDIR *opendir(name)char *name;
X{
X
X    DIR *directory;
X    int status;
X    char *p;
X
X    if((directory=MALLOC(DIR))==NULL)
X         return NULL;
X
X    directory->FileFlags=D_FIRST;
X
X    /* parse and construct full path name */
X    if(index(name,':')!=NULL)
X         strcpy(directory->FilePath,name);
X    else
X    {
X         status=Dgetdrv();
X         directory->FilePath[0]=status+'A';
X         directory->FilePath[1]=':';
X         Dgetpath(directory->FilePath+2,status+1);
X         strcat(directory->FilePath,"\\");
X         strcat(directory->FilePath,name);
X    }
X
```

```
X    for(p=directory->FilePath;*p;p++);
X         if(*p=='/') *p='\\';
X    if(index(directory->FilePath,'*') || index(directory->FilePath,'?'))
X         ;
X    else
X    {
X         p=directory->FilePath;
X         if(p[strlen(p)-1])
X              strcat(p,"\\*.*");
X         else
X              strcat(p,"*.*");
X    }
X    directory->FileDirect.d_ino=0;
X    return directory;
X}
X
Xstatic struct direct *getdent(dir)DIR *dir;
X{
X    struct direct *d_ans= &(dir->FileDirect);
X
X    strcpy(d_ans->d_name,dir->FileName);
X    d_ans->d_namlen=strlen(d_ans->d_name);
X    d_ans->d_reclen=sizeof(struct direct);
X    d_ans->d_ino++;
X    return d_ans;
X}
X
Xstruct direct *readdir(dir)DIR *dir;
X{
X    int res;
X    DMABUFFER *hold;
X
X    struct direct *d_struct= &(dir->FileDirect);
X    hold=Fgetdta();
X    Fsetdta(dir);
X    if(dir->FileFlags & D_FIRST)
X    {
X         res=Fsfirst(dir->FilePath,0xff);
X         dir->FileFlags &= ~(D_FIRST);
X    }
X    else
X         res=Fsnext();
X    Fsetdta(hold);
X    if(!res)
X         return getdent(dir);
X    else return NULL;
X}
X
Xvoid closedir(dir)DIR *dir;
X{
X    free(dir);
X}
X
```

```
Xvoid rewinddir(dir)DIR *dir;
X{
X    dir->FileFlags=D_FIRST;
X    dir->FileDirect.d_ino=0;
X}
X
Xvoid seekdir(dir,where)DIR *dir;long where;
X{
X    struct direct *w;
X    rewind(dir);
X    while(dir->FileDirect.d_ino!=(where-1))
X         w=readdir(dir);
X}
X
Xlong telldir(dir)DIR *dir;
X{
X    return (long)(dir->FileDirect.d_ino);
X}
XSHAR_EOF
Xchmod +x 'ndir.c'
Xfi # end of overwriting check
Xif test -f 'ndir.h'
Xthen
X    echo shar: will not over-write existing file "'ndir.h'"
Xelse
Xcat << \SHAR_EOF > 'ndir.h'
X/* @(#)ndir.h     1.4  4/16/85 */
X/* Modified for the atari st series 5/4/87 */
X#ifndef DEV_BSIZE
X#define  DEV_BSIZE 512
X#endif
X#define DIRBLKSIZ DEV_BSIZE
X#define  MAXNAMLEN 255
X
Xstruct  direct {
X    long d_ino;             /* inode number of entry */
X    short    d_reclen;      /* length of this record */
X    short    d_namlen;      /* length of string in d_name */
X    char d_name[MAXNAMLEN + 1]; /* name must be no longer than this */
X};
X
X/*
X * The DIRSIZ macro gives the minimum record length which will hold
X * the directory entry.  This requires the amount of space in struct direct
X * without the d_name field, plus enough space for the name with a terminating
X * null byte (dp->d_namlen+1), rounded up to a 4 byte boundary.
X */
X
X#ifdef DIRSIZ
X#undef DIRSIZ
X#endif
X#define DIRSIZ(dp) \
X    ((sizeof (struct direct) - (MAXNAMLEN+1)) + (((dp)->d_namlen+1 + 3) &~ 3))
```

```
X
X/*
X * Definitions for library routines operating on directories.
X */
X
X/* Structure of a directory entry under GEMDOS */
X
Xtypedef struct _dirdesc
X  {
X    char Reserved[21];
X    char FileAttr;
X    int      CreateTime;
X    int      CreateDate;
X    long FileSize;
X    char FileName[14];
X    char FilePath[256];
X    int      FileFlags;
X    struct direct FileDirect;
X  } DIR;
X
X#ifndef NULL
X#define NULL 0L
X#endif
X/* values of FileFlags */
X#define D_NONE 0x00
X#define D_FIRST 0x01
X
X#ifndef DEF_NDIR
Xextern  DIR *opendir();
Xextern  struct direct *readdir();
Xextern  void closedir();
Xextern  void seekdir();
Xextern long telldir();
Xextern void rewinddir();
X#endif
X
XSHAR_EOF
Xchmod +x 'ndir.h'
Xfi # end of overwriting check
Xif test -f 'ndir.man'
Xthen
X    echo shar: will not over-write existing file "'ndir.man'"
Xelse
Xcat << \SHAR_EOF > 'ndir.man'
X
X    NAME ndir - portable directory reading routines
X
X    SUMMARY
X
X    #include <ndir.h>
X
X
X    DIR *opendir(name)
```

```
X        char *name;
X
X    struct direct *readdir(dirp)
X        DIR *dirp;
X
X    void closedir(dirp);
X        DIR *dirp;
X
X    void seekdir(dirp,where)
X        DIR *dirp;
X        long where;
X
X    long telldir(dirp);
X        DIR *dirp;
X
X
X    Opendir() opens a dirctory for reading and returns a pointer to a DIR
Xfor this directory.
X
X    readdir() reads the next entry in the directory and returns a pointer
Xto a struct direct for this readding.
X
X    closedir() closes the directory.
X
X    seekdir() sets directory read to a given place in the directory.
X
X    telldir() returns the place of the next read in the directory.
X
X    struct direct is defined in the header file as:
X
X    struct direct
X    {
X        long d_ino;        /* inode number of entry */
X        short d_reclen;        /* length of this record */
X        short d_namlen;        /* length of the string d_name */
X        char d_name[MAXNAMLEN+1];/*basename -must not be longer than
X                        MAXNAMLEN */
X    };
XSHAR_EOF
Xchmod +x 'ndir.man'
Xfi # end of overwriting check
Xif test -f 'scandir.c'
Xthen
X    echo shar: will not over-write existing file "'scandir.c'"
Xelse
Xcat << \SHAR_EOF > 'scandir.c'
X/*
X**  SCANDIR
X**  Scan a directory, collecting all (selected) items into a an array.
X*/
X#ifdef MWC
X#include <types.h>
X#include "ndir.h"
```

```
X#else
X#include <sys/types.h>
X#include <sys/dir.h>
X#endif
X
X#ifdef  RCSID
Xstatic char RCS[] = "$Header: scandir.c,v 1.1 87/12/29 21:35:56 rsalz Exp $";
X#endif  /* RCSID */
X
X/* Initial guess at directory size. */
X#define INITIAL_SIZE  20
X
X/* A convenient shorthand. */
Xtypedef struct direct  ENTRY;
X
X/* Linked in later. */
Xextern char      *malloc();
Xextern char      *realloc();
Xextern char      *strcpy();
X
X
Xint
Xscandir(Name, List, Selector, Sorter)
X    char       *Name;
X    ENTRY         ***List;
X    int            (*Selector)();
X    int            (*Sorter)();
X{
X    register ENTRY     **names;
X    register ENTRY      *E;
X    register DIR    *Dp;
X    register int     i;
X    register int     size;
X
X    /* Get initial list space and open directory. */
X    size = INITIAL_SIZE;
X    if ((names = (ENTRY **)malloc(size * sizeof names[0])) == NULL
X     || (Dp = opendir(Name)) == NULL)
X    return(-1);
X
X    /* Read entries in the directory. */
X    for (i = 0; E = readdir(Dp); )
X    if (Selector == NULL || (*Selector)(E)) {
X        /* User wants them all, or he wants this one. */
X        if (++i >= size) {
X         size <<= 1;
X         names = (ENTRY **)realloc((char *)names, size * sizeof names[0]);
X         if (names == NULL) {
X             closedir(Dp);
X             return(-1);
X         }
X        }
X
```

```
X        /* Copy the entry. */
X        if ((names[i - 1] = (ENTRY *)malloc(DIRSIZ(E))) == NULL) {
X         closedir(Dp);
X         return(-1);
X        }
X        names[i - 1]->d_ino = E->d_ino;
X        names[i - 1]->d_reclen = E->d_reclen;
X        names[i - 1]->d_namlen = E->d_namlen;
X        (void)strcpy(names[i - 1]->d_name, E->d_name);
X    }
X
X    /* Close things off. */
X    names[i] = NULL;
X    *List = names;
X    closedir(Dp);
X
X    /* Sort? */
X    if (i && Sorter)
X    qsort((char *)names, i, sizeof names[0], Sorter);
X
X    return(i);
X}
XSHAR_EOF
Xchmod +x 'scandir.c'
Xfi # end of overwriting check
Xif test -f 'scandir.man'
Xthen
X    echo shar: will not over-write existing file "'scandir.man'"
Xelse
Xcat << \SHAR_EOF > 'scandir.man'
X
X    NAME scandir, alphasort - scan a directory
X
X    SYNOPSIS
X
X#include <types.h>
X#include <ndir.h>
X
Xint
Xscandir(name, list, selector, sorter)
X    char *name;
X    struct direct ***list;
X    int (*selector)();
X
X
Xint (*sorter)();
X
X
Xint
Xalphasort(d1, d2)
X    struct direct **d1;
X    struct direct **d2;
X
```

```
X    DESCRIPTION
X
X    Scandir reads the directory name and builds a NULL\-terminated array
Xof pointers to the entries found in that directory.  This array is put into the
Xlocation pointed to by the list parameter.
X
X    If the selector parameter is non-NULL, it is taken to be a pointer to
Xa function called with each entry, to determine whether or not it should be
included in
Xthe returned list.  If the parameter is NULL, all entries are included.
X
X    As an added feature, the entries can be sorted (with qsort (3))
Xbefore the list is returned.  If the sorter parameter is non-NULL, it is passed
Xto qsort to use as the comparison function.  The alphasort routine is provided
Xto sort the array alphabetically.
X
X    The array pointed to by list and the items it points to are all space
Xobtained through malloc (3), and their storage can be reclaimed as shown in the
Xexample below.
X
X    "EXAMPLE"
X
X    Here is a small ls (1)\-like program:
X
X#include <stdio.h>
X#include <types.h>
X#include <stat.h>
X#include <ndir.h>
X
Xextern int alphasort();
X
Xstatic int
Xfilesonly(e)
X    struct direct *e;
X{
X    struct stat sb;
X
X    return(stat(e->d_name, &sb) >= 0 && (sb.st_mode & S_IFMT) == S_IFREG);
X}
X
Xmain(ac, av)
X    int ac;
X    char *av[];
X{
X    register int i;
X    register int j;
X    struct direct **list;
X
X    if (ac != 2) {
X         fprintf(stderr, "usage: %s dirname\n", av[0]);
X         exit(1);
X    }
X    if (chdir(av[1]) < 0) {
```

```
X        perror(av[1]);
X        exit(1);
X    }
X    if ((i = scandir(".", &list, filesonly, alphasort)) < 0) {
X        perror("Error reading directory");
X        exit(1);
X    }
X    for (j = 0; j < i; j++)
X        printf("%s\n", list[j]->d_name);
X    for (j = 0; j < i; j++)
X        free((char *)list[j]);
X    free((char *)list);
X    exit(0);
X}
X
X
X    "SEE ALSO"
X    ndir(3), qsort(3)
X
X    DIAGNOSTICS
X
X    Returns the number of entries in the ``list,'' or -1 if the directory
Xcould not be opened or a memory allocation failed.
X
X
X    BUGS
X    The routine can be slightly wasteful of space.
XSHAR_EOF
Xchmod +x 'scandir.man'
Xfi # end of overwriting check
X#   End of shell archive
Xexit 0
SHAR_EOF
chmod 0600 stftw.shr || echo "restore of stftw.shr fails"
exit 0


From -2438460064617039625
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,7188660b375b29ad,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-12-19 13:58:43 PST
Path: utzoo!utgpu!watmath!clyde!att!osu-cis!tut.cis.ohio-state.edu!rutgers!ucsd!
nosc!helios.ee.lbl.gov!pasteur!agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i078: raymovi2 -- Raytrace movie generator  part01/02
Keywords: shar, Alcyon, low
Message-ID: <5799@saturn.ucsc.edu>
Date: 19 Dec 88 21:58:43 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1286
Approved: koreth@ssyx.ucsc.edu
```

Submitted-by: aking@bbn.com (Allen King)
Posting-number: Volume 1, Issue 78
Archive-name: raymovi2/part01

RAYMOVI.PRG

RAYMOVI.PRG renders 3-dimensional scenes in a world of reflective and refractive spheres, using ray tracing techniques. Animated sequences of frames can be generated, showing reflective balls linked and bouncing in a gravitational field. They resemble a string of bouncing pearls.

RAYMOVI.PRG runs only on a color ST in low resolution. 1 Meg is required to generate images with resolutions greater than one ray per pixel or for replay of movies from RAM. Operation with only 512K is possible, but will produce slightly aliased pictures (one ray per pixel) which can only be displayed statically, slowly from floppy, or from hard disk (if you have a disk blaster program).

The files in RAYMOV2S.SHAR, the source sharfile, are:

```
C Sources:    BLASTRAM.C, MAIN.C, WNDOWS.C, FILE.C, MOVIE.C, SHADE.C,
         FIND.C, SUPPORT.C
Include files:     RTD.H, STYLE.H, COLORS.H, EXTERN.H
Pgm Generation: MAKEFILE, BLASTRAM.LNK, RAYMOVI.LNK
RAYMOVI.DOK   documentation
PEARLS        the numeric description of an interesting scene
```

RAYMOVI was last compiled with ALCYON. The MAKEFILE and *.LNK are (therefore) pretty crufty. Use them accordingly.

RAYMOVI requires an 8K stack. Modify your copy of gemstart.s (or whatever defines your stack size) appropriately.

Allen King aking pebbles.bbn.com ucbvax
30 Gibson St 1-617-449-3359 evenings
Needham Ma.
02192


```
#------------------------------ cut here ------------------------------
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
# This is part 1 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
#
#    Run the following text with /bin/sh to create:
#      BLASTRAM.C
#      BLASTRAM.LNK
#      COLORS.H
```

```
#      EXTERN.H
#      FILE.C
#      FIND.C
#      MAIN.C
#      MAKEFILE
#      MOVIE.C
#      PEARLS
#      RAYMOVI.DOK
#      RAYMOVI.LNK
#      README
#      RTD.H
#      SHADE.C
#      STYLE.H
#      SUPPORT.C
#      WNDOWS.C
#
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
sed 's/^X//' << 'SHAR_EOF' > BLASTRAM.C &&
X#include <osbind.h>
X#include <stdio.h>
X
Xint XMIN, YMIN, XMAX, YMAX;
X
XwOnewSize(x,y, w,h)
Xint x,y, w,h;
X{   XMIN = x; YMIN = y;
X    XMAX = x+w+2; YMAX = y+h+2;
X}
X
X#define colors(r,g,b) 1000*(2*(r)+1)/16, 1000*(2*(g)+1)/16, 1000*(2*(b)+1)/16
Xint oldClut[16][3], newClut[16][3] = {
Xcolors(7,7,7),    /* all white-ish */
Xcolors(0,0,0),    /* black */
Xcolors(0,0,1),    /* somewhat lighter */
Xcolors(1,1,1),
Xcolors(1,1,2),
Xcolors(2,2,2),
Xcolors(2,2,3),
Xcolors(3,3,3),
Xcolors(3,3,4),
Xcolors(4,4,4),
Xcolors(4,4,5),
Xcolors(5,5,5),
Xcolors(5,5,6),
Xcolors(6,6,6),
Xcolors(6,6,7),
Xcolors(6,7,7)};   /* almost white */
X
Xint wHandle;
XFILE *fp, *fopen();
```

```
Xchar outname[] = "blastram";
Xlong orig_base;
Xfloat speed = 12000.0;
Xlong Ispeed = 12000;
X
X#define bytesPscreen 32000
X
Xmain (argc, argv)
Xint argc;
Xchar *argv[];
X{   long mem_avail, l;
X    char *mem_for_screen;
X    char (*Pscreen)[bytesPscreen];
X    int i, j, Nscreens, screens_read, delay;
X    char fname[20];
X
X    if (argc != 2)
X         panic("usage: blastram <fname_prefix> (sans the XXX.PI1)\n");
X    wHandle = w_open(0, "pearle", 0);     /* open window (typeless, menueless) */
X    w_lClut(newClut, oldClut);
X    v_hide_c(wHandle);
X    orig_base = Physbase();
X
X    mem_avail = Malloc((long)-1);
X    mem_for_screen = Malloc (mem_avail);
X    printf("\033E%ld bytes memory available.\n", mem_avail);
X
X    Nscreens = (mem_avail - 0x100 - ((long)mem_for_screen & 0xff))
X                                  / bytesPscreen;
X    Pscreen = (char *) (((long)mem_for_screen | 0xff) + 1);
X
X    printf("%d screens Max.\n\n", Nscreens);
X
X    for (i=1; i<999 & Nscreens > 0;)
X    {    sprintf(fname,"%s%03d.pi1", argv[1], i);
X         printf("file %s -- ", fname);
X         if (f_read(fname, Pscreen+screens_read) >= 0)
X         {    screens_read++;
X              --Nscreens;
X              printf("to screen %d.\n", screens_read);
X              i++;
X         }
X         else
X         {    printf("not found.\n");
X              if (screens_read)
X              {    char key;
Xreask:                 printf("\nType 'c', 'd' or 'e':\n");
X                   printf("c: CONTINUE with another diskette\n");
X                   printf("d: DISPLAY what's been loaded\n");
X                   printf("e: EXIT\n");
X                   key = Cconin();
X                   printf("\n");
X                   Mediach('a'-'a');
```

```
X                 if (key == 'c')
X                      continue;
X                 else if (key == 'd')
X                      break;
X                 else if (key == 'e')
X                      exit(0);
X                 else
X                 {    printf("\nAnswer 'c', 'd', or 'e'\n\n");
X                      goto reask;
X    }    }    }    }
X    for (i=0; 1; i = (i + kbd() + screens_read)%screens_read)
X    {    Setscreen((long)-1, Pscreen + i, -1);
X         for (l=0; l < Ispeed; l++)
X              delay++;
X}   }
X
Xint freeze = 0;
X
Xkbd()
X{   char key, fname[20];
X    int found_first = 0, fr1;
X
X    if (Cconis() == 0 && freeze==0)
X         return(1);
X
X    key = Cconin();
X    while (1)
X    {    switch(key)
X         {   case ('?'):    /* help */
X              Setscreen((long)-1, orig_base, -1);
X              printf("\033E\n\n       BLASTRAM.C\n\
X              \n '-' back up one frame\
X              \n '+' forward one frame\
X              \n ' ' freeze on frame\
X              \n 'f' faster\
X              \n 's' slower\
X              \n <all else> play forward\
X              \n use reset to exit\
X              \n");
X              key = Cconin();
X              printf("\033E");
X              break;
X             case ('-'):                 /* back up one */
X              freeze = 1;
X              return (-1);
X             case ('+'): case ('='):     /* forward one */
X              freeze = 1;
X              return (1);
X             case (' '):                 /* freeze on */
X              freeze = 1;
X              return (0);
X             case ('f'):                 /* faster */
X              speed /= 1.2;
```

```
X             Ispeed = speed;
X             return (1);
X            case ('s'):                  /* slower */
X             speed *= 1.2;
X             Ispeed = speed;
X             return (1);
X/*          case ('e'):
X             Setscreen((long)-1, orig_base, -1);
X             w_lClut(oldClut, (long)0);
X             w_close();
X             exit(0);/* the above leaves desktop in a wierd state */
X            default:
X             freeze = 0;
X             return (1);
X}   }    }
X
Xpanic(s,l1,l2)
Xchar *s;
Xlong l1,l2;
X{   printf("\n***** FATAL ERROR *****\n");
X    printf(s,l1,l2);
X    printf("\nPress any key to continue\n");
X    Cconin();
X    exit();
X    atoi();
X}
SHAR_EOF
chmod 0600 BLASTRAM.C || echo "restore of BLASTRAM.C fails"
sed 's/^X//' << 'SHAR_EOF' > BLASTRAM.LNK &&
X\include\gemstart.o blastram.o file.o wndows.o
X\include\aesbind \include\vdibind \include\gemlib \include\osbind.o
X\include\libf
SHAR_EOF
chmod 0600 BLASTRAM.LNK || echo "restore of BLASTRAM.LNK fails"
sed 's/^X//' << 'SHAR_EOF' > COLORS.H &&
X/* colors.h  defines palletes
X
XOBJECTS:|    cr   cg   cb
X----------------+-------------------------------
Xballs:       |
X   white|    1    1    1
X   red       |    1    0    0
X        |
Xfloor:       |
X   black cracks   |    0    0    0
X   white tiles    |    1    1    1
X   blue tiles     |    0    0    1
X
X
XMODE (#colors)          generated intensities |    color number output
Xind:              cr   cg   cb   |    white    red  blue
X-------------------------------------------------+-------------------------------
X1            -ign-    -ign-    x    |    1-15,0    -
```

```
X2            x    -ign-     x    |    1-6,0     8-15
X3            x    x    x    |    1-5,0     1,6-10    1,11-15
X
X/**/
X             /* ind = 1: 1 color */
Xcolors(7,7,7)     /* all white-ish */
Xcolors(0,0,0)     /* black */
Xcolors(0,0,1)     /* somewhat lighter */
Xcolors(1,1,1)
Xcolors(1,1,2)
Xcolors(2,2,2)          /* note: since there are only 8 grey scales in the */
Xcolors(2,2,3)          /*       hardware (per color) and we generate 16   */
Xcolors(3,3,3)          /*    some hue variation will be produced as
        */
Xcolors(3,3,4)          /*    the intensity varies
        */
Xcolors(4,4,4)
Xcolors(4,4,5)
Xcolors(5,5,5)
Xcolors(5,5,6)
Xcolors(6,6,6)
Xcolors(6,6,7)
Xcolors(6,7,7)     /* almost white */
X
X             /* ind = 2: 2 color */
Xcolors(7,7,7)     /* white */
Xcolors(0,0,0)
Xcolors(1,1,1)
Xcolors(2,2,2)
Xcolors(3,3,3)
Xcolors(4,4,4)
Xcolors(5,5,5)
Xcolors(6,6,6)
X
Xcolors(0,0,0)     /* red */
Xcolors(1,0,0)
Xcolors(2,0,0)
Xcolors(3,0,0)
Xcolors(4,0,0)
Xcolors(5,0,0)
Xcolors(6,0,0)
Xcolors(7,0,0)
X
X             /* ind = 3: 3 color */
Xcolors(7,7,7)     /* white */
Xcolors(0,0,0)
Xcolors(2,2,2)
Xcolors(4,4,4)
Xcolors(5,5,5)
Xcolors(6,6,6)
X
Xcolors(3,0,0)     /* red */
Xcolors(4,0,0)
```

```
Xcolors(5,0,0)
Xcolors(6,0,0)
Xcolors(7,0,0)
X
Xcolors(0,0,3)     /* blue */
Xcolors(0,0,4)
Xcolors(0,0,5)
Xcolors(0,0,6)
Xcolors(0,0,7)
SHAR_EOF
chmod 0600 COLORS.H || echo "restore of COLORS.H fails"
sed 's/^X//' << 'SHAR_EOF' > EXTERN.H &&
Xextern struct ball bl[];
Xextern struct sphere ls;
Xextern int level,nob;
X
X
SHAR_EOF
chmod 0600 EXTERN.H || echo "restore of EXTERN.H fails"
sed 's/^X//' << 'SHAR_EOF' > FILE.C &&
X#include "\include\gemdefs.h"
X#include "\include\osbind.h"
X#include "rtd.h"
Xint debug;
X
X#define GRMODE   0         /* graphics mode 0 (320x200) */
X#define FSIZE  32034       /* size of DEGAS save-file */
Xint vs_handle;         /* virtual screen (workstation) handle */
Xint *clut_fetch();
Xint read_pal[16];
X
X/* the first FSIZE bytes of the file is DEGAS */
X
X/* save screen data to disk in DEGAS-like format */
X
Xf_write(fn)
Xchar *fn;
X{
X   char *scrnp;
X   int fd, n;
X   short gmode[1];
X
X   if (fn == 0)
X    return (-1);
X   scrnp = (char *)Physbase() ;  /* get pointer to display frame */
X   *gmode = GRMODE ;
X
X   if ((fd = creatb(fn, 0755)) < 0)
X    return(-1);
X
X   n = write(fd, gmode, 2);          /* write graphics mode */
X   n += write(fd, clut_fetch() /*palet(ind)*/, 32);    /* write palette */
X   n += write(fd, scrnp, 32000);     /* write screen data */
```

```
X   close(fd);
X
X   if (n != FSIZE)
X   {  unlink(fn);
X      panic("Not enough space on disk to write file\n(need 32K)");
X   }
X   return(0) ;
X}
X
X/* read file (DEGAS format) */
Xint f_read(fn, buf)
Xchar *fn, *buf;
X{
X   char *scrnp;
X   int fd, n;
X   short gmode[1];
X
X   if (fn == 0)
X    return (-1);
X   if ((fd = openb(fn, 0)) < 0)
X    return(-1);
X
X   n = read(fd, gmode, 2);      /* read graphics mode */
X   n += read(fd, &read_pal, 32);    /* read palette */
X   n += read(fd, (buf? buf: (char *)Physbase()), 32000);
X   close(fd);                   /* read screen data */
X
X   if (n != FSIZE)
X    return(-2);            /* file read error: file too short */
X   clut_put(&read_pal);
X
X
X   return(0);
X}
X
Xint scramble[16] = {0, 2, 3, 6, 4, 7, 5, 8, 9, 10, 11, 14, 12, 15, 13, 1};
X
Xint *clut_fetch()
X{   int *p = read_pal, i;
X    int rgb[3], red, green, blue;
X
X    for (i=0; i<16; i++)
X    {    vq_color(vs_handle, scramble[i], 0, rgb);
X         red   = rgb[0] / 125;
X         green = rgb[1] / 125;
X         blue  = rgb[2] / 125;
X         if (debug)
X              printf("cF(%d>%d): %d %d %d ->%d %d %d\n",i,scramble[i],
X                   rgb[0], rgb[1], rgb[2], red, green, blue);
X         *p++ = (red*0x100 + green*0x10 + blue);
X    }
X    return(read_pal);
X}
```

```
X
X#define clr(r) 1000*(2*(r)+1)/16;
X
Xclut_put(p)
Xint *p;
X{   int i;
X    int rgb[3], red, green, blue;
X    for (i=0; i<16; i++, p++)
X    {    rgb[0] = clr( *p >>8 & 0xf);     /* red */
X         rgb[1] = clr( *p >>4 & 0xf);     /* green */
X         rgb[2] = clr( *p & 0xf);    /* blue */
X         vs_color(vs_handle, scramble[i], rgb);
X}   }
SHAR_EOF
chmod 0600 FILE.C || echo "restore of FILE.C fails"
sed 's/^X//' << 'SHAR_EOF' > FIND.C &&
X#include <math.h>
X#include "rtd.h"
X#include "extern.h"
X
X/* find where a ray leaves a sphere */
X
Xdouble  findo (m, s)
X    struct mat *m;
X    struct sphere  *s;
X{
X    struct vector   foops;
X    double  t;
X
X    /* foops is the rotated vector for the center of the sphere
X       with respect to the beginning of the ray */
X
X    mt_vec (&foops, m, &(s->cent));
X
X    /* find out if the center of the sphere is within range.
X        (it should be for this...)*/
X
X    t = s->rad * s->rad - foops.y * foops.y - foops.z * foops.z;
X
X    /*return distance from original entry. we can find the point
X       when we get out*/
X
X    if (t > 0)
X    t = foops.x + sqrt (t);
X    else
X    t = 0;
X
X    return (t);
X}
X
X
X/* find where a ray next hits (enters or exits) a sphere */
X
```

```
Xdouble  findx (m, s)
X    struct mat *m;
X    struct sphere  *s;
X{
X    struct vector   foops;
X    double  t;
X
X    /* foops is the rotated vector for the center of the sphere
X       with respect to the beginning of the ray */
X    mt_vec (&foops, m, &s->cent);
X    t = s->rad*s->rad - foops.y*foops.y - foops.z*foops.z;
X    if (t < 0)
X    t = HUGE;        /* doesn't touch, return flag */
X    else
X    {   t = sqrt(t);
X    if (foops.x > t + EPSILON)
X         t = foops.x - t;   /* return distance to entering */
X    else if (foops.x > - t + EPSILON)
X         t = -foops.x - t;  /* return minus distance to exiting */
X    else
X         t = HUGE;
X    }
X    return (t);
X}
X
X/* see above. the only difference is that
Xthe value returned is foops.x - sqrt(t) */
X
Xdouble  find (m, s)
X    struct mat *m;
X    struct sphere  *s;
X{
X    struct vector   foops;
X    double  t;
X
X    mt_vec (&foops, m, &(s->cent));
X
X    t = s->rad * s->rad - foops.y * foops.y - foops.z * foops.z;
X
X    if (t > 0)
X    t = foops.x - sqrt (t);
X    else
X    t = 0;
X
X    return (t);
X}
X


X
X/* returns value telling how much a sphere
Xis occluding the light source */
X
Xdouble  finds (m, s)
```

```
X    struct mat *m;
X    struct sphere  *s;
X{
X    struct vector   foops;
X    double  t;
X
X    mt_vec (&foops, m, &(s->cent));
X
X    t = s->rad - sqrt (foops.y * foops.y + foops.z * foops.z);
X
X    if (t > 0)
X    t = t / foops.x;
X    else
X    t = 0;
X
X    return (t);      /* radians between ray and edge of intersecting sphere*/
X}
X
X

X/* gets amount of diffuse light hitting a point */
X
Xfloat   shadow (p, rd, gn, blu)
Xstruct vector  *p;
Xdouble *rd, *gn, *blu;
X{
X    struct mat  trans;
X    struct sphere   ss;
X    struct vector   d;
X    int     c,
X            i;
X    double  l,
X            k,
X            x,
X            y,
X            z,
X            finds ();
X
X    l = 0.0;
X    c = -1;
X    sv (&d, &(ls.cent), p);
X    vecl (&d);  /*!!*/
X    vexzl (&d); /*!!*/
X    mt (&(d), &trans);
X
X    /* get maximum obscurment */
X
X    for (i = 0; i < nob; i++) {
X    ss.rad = bl[i].s.rad;
X    sv (&(ss.cent), &(bl[i].s.cent), p);
X    if ((k = finds (&trans, &ss)) > l) {
X        c = i;
X        l = k;
```

```
X    }
X    }
X
X    if (c == -1)
X    k = 255.0;
X    else {
X    k = 1.0 - l * d.l / ls.rad;
X    if (k < 0.0)  /* l           = angle to move to get ray out of sphere*/
X        k = 0.0;  /* d.l/ls.rad = 1/angle light source subtends */
X    else
X        k *= 255.0;
X    }
X    *rd = *gn = *blu = k;  /* all colors the same for now */
X}
X
X
SHAR_EOF
chmod 0600 FIND.C || echo "restore of FIND.C fails"
sed 's/^X//' << 'SHAR_EOF' > MAIN.C &&
X/* main.c
X *
X ****************************************************************
X *                                                              *
X *                      basic ray tracing:                      *
X *    spheres and a floor (reflection, refraction and diffuse)  *
X *      programmer: friedrich knauss              *
X *                       7-4-86 to 7-21-86                      *
X *      ported to st: allen king 5/87, added:     *
X *           limited color               *
X *           low rez display first            *
X *           .25 to 16 rays per pixel         *
X *           object definitions in file       *
X *           multiple-frame movies            *
X *           bouncing and gravity             *
X *                                                              *
X ****************************************************************/
X
X#include <stdio.h>
X#include <math.h>
X#include <osbind.h>
X#include <gemdefs.h>
X#include "rtd.h"
X#define dprintf if (debug) printf
X
X/* these definitions describe a window in the x-y plane
X   that the whole thing is viewd through. */
X
Xint XMIN, YMIN, XMAX, YMAX;
Xfloat xmin, ymax;     /* floating pt versions */
X
XwOnewSize(x,y, w,h)
Xint x,y, w,h;
X{   XMIN = x; YMIN = y;
```

```
X    XMAX = x+w+2; YMAX = y+h+2;
X}
X
X/* Nomenclature notes:
X *  1. <a>P<b> stands for "the number of <a>'s PER <b>
X *  2. r stands for rays
X *  3. p stands for pixels
X *  4. s stands for either side of the screen
X *  5. the suffix "Max" denotes the max number compiled into the program
X *
X * Note: these are linear (not area) conversions. A value of 2 rPp (rays PER
X * pixel) is applied in both x and y directions to get an area (and more
X * true to life) conversion of 4 (square) rays through each (square) pixel.
X */
X    float rPp;             /* rays PER pixel (on a (linear) side)*/
X#   define rPpMax 4
X                  /* maximum rays PER pixel (ditto) */
X    int pPs;           /* pixels PER side (of screen) */
X    int rPs;           /* rays PER side (of screen) */
X    int rPsMax;            /* Max rays PER side (of screen) */
X
X#define INC  1
X#define INCY  (1.25/rPpMax)
X#define INCX  (1.0/rPpMax)
X
Xint colorInd;              /* color index (1->b/w, 2->2tones, 3->3tones)*/
X
X#define colors(r,g,b) 1000*(2*(r)+1)/16, 1000*(2*(g)+1)/16, 1000*(2*(b)+1)/16,
Xint oldClut[16][3], trClut[][3] =
X{
X#include "colors.h"
X0};
X
Xstruct
X{   int (*pal)[][3];
X    int Navg;
X} cConfig[] = {{trClut, 16}, {trClut+16, 32}, {trClut+32, 43}};
X
XFILE *fp, *fopen();
Xchar getline();
Xchar outname[6] = "pearl";
Xint wHandle;
Xint pxy_array[4];
X
X#define Nball 15
Xstruct ball bl[Nball];
Xint     level,
X        nob;
Xstruct vector vp;
Xfloat gravity = 0.0, attraction = 0.0;
Xint step, Step = 1000, bounce = 1;;
Xint fr=0, frames=1;
X
```

```
Xint debug =0;
X
Xstruct sphere   ls;
X

Xmain ()
X{   static float    xco,
X                    yco, yup;
X    struct ray  rr;
X    int     h, i, j, k;
X    int c, cr, cg, cb;
X    int x0, y0, x1, y1, dx, dy, dmax, xMax, yMax;
X    double red, green, blue;                  /* was int */
X    long ctr, BavgN;
X    float tmp;
X
X    char str[200], *buf;
X    int Navg, *avg, *Bavg;
X
X    rPp = 1.0;/* defaults: */
X    colorInd = 1;
X
X/*    graf_mouse(M_OFF, 0L);/**/
X    printf("\033E");
X    do
X    {   printf("                RAYMOVI.PRG\n\n");
X    printf("Enter scene descriptor\nfilename: ");
X    if (getline(str, sizeof(str)) == '\003')
X        exit(0);
X    printf("\nreading file %s\n", str);
X    } while(scene_inz(str) == 0);
X
X    printf("\033E");
X
X    wHandle = w_open(0, "raymovi", 0);   /* open window (typeless, menueless) */
X    w_lClut(cConfig[colorInd-1].pal, oldClut);
X    v_hide_c(wHandle);
X
X    xmin = XMIN;  ymax = YMAX;
X    dx = XMAX-XMIN;
X    dy = YMAX-YMIN;
X    if (rPp > 1.0001)
X    {   BavgN = (long)dx * dy * colorInd;
X    Bavg = Malloc(BavgN * sizeof(int));
X    if (Bavg == 0)
X    {   long l1;
X        printf("limited to 1 ray/pixel due to insufficient RAM\n");
X        if (rPp > 1.0)
X         rPp = 1.0;
X        for (l1=0; l1<200000; l1++);
X    }
X    for (tmp = 1.0; rPp >= 1.0001; tmp *= 2.0, rPp /= 2.0);
X    }
```

```
X    else
X    for (tmp = 1.0; rPp < .5001; tmp /= 2.0, rPp *= 2.0);
X    rPp = tmp;
X    dmax = (dx>dy)? dx: dy;
X    for (i=dmax-1,       pPs=1;     i>0   ; i>>=1, pPs<<=1);
X    for (i=rPp*pPs-0.5,    rPs=1;   i>0   ; i>>=1, rPs<<=1);
X    rPsMax = rPpMax * pPs;
X    xMax = ((long)rPsMax*dx)/pPs;    yMax = ((long)rPsMax*dy)/pPs;
X
X    for (; fr<=frames; fr++)
X    {   int len, len1, p34, x, y, Elen, Ey;
Xredraw:
X    if (Bavg) for (avg = Bavg, ctr=0; ctr<BavgN; ctr++)
X         *avg++ = 0;
X    Navg = cConfig[colorInd-1].Navg;
X    ctr = 0;
X    for (len = rPsMax; len>= rPsMax/rPs; len /=2)
X    {   p34 = 3;
X        if (len>=rPpMax)
X        {    len1 = len/rPpMax - 1;
X         Elen = 0;
X        }else
X        {    len1 = 0;
X            Elen = rPpMax -1;
X         Navg *= 4;
X        }
X        for (y=0; y<yMax; y += len)
X        {    p34 = (p34 | 1) ^ 2;
X         if (Elen)
X             Ey = (((y + len)&Elen) == 0);
X         y0 = y/rPpMax;
X         y1 = y0 + YMIN;
X         pxy_array[1] = y1;
X         pxy_array[3] = y1 + len1;
X         yco = ymax - INCY*y;
X
X         for (x=0; x<xMax; x += len)
X         {   if ((p34 ^=1) == 0 && ctr !=0)
X              continue;
X             ctr++;
X             x0 = x/rPpMax;
X             x1 = x0 + XMIN;
X             pxy_array[0] = x1;
X             pxy_array[2] = x1 + len1;
X             xco = xmin + INCX*x;
X
X            /* define the ray through a pixel; find out value for that pixel */
X             mv (xco, yco, 0.0, &(rr.org));
X             sv (&(rr.dir), &(rr.org), &vp);
X             shade (&rr, &red, &green, &blue);   /* find out color */
X
X             cb = blue;
X             cr = red;
```

```
X            cg = green;
X            if (Bavg)      /* compute sum for all rays */
X            {   register int *array;
X             array = Bavg + ((long)y0*dx+x0+1)*colorInd;
X             switch(colorInd)
X             { case 3:
X                 cg = *--array += cg;
X               case 2:
X                 cr = *--array += cr;
X               case 1:
X                 cb = *--array += cb;
X            }   }
X
X            if (Elen==0 || Ey && ((x+len)&Elen) == 0 )
X            {    switch(colorInd)
X             { case 1:          /* white (and black): */
X                 c = (cb/Navg + 1) & 0xf;         /* white */
X                 break;
X               case 2:          /* white and red: */
X                 if (2*cr < 3*cb)
X                  c = cb/Navg+1 & 7;          /* white */
X                 else c = (cr/Navg & 7) + 8;      /* red */
X                 break;
X               case 3:          /* white, red, blue: */
X                 if (4*cr > 3*(cg+cb))
X                 {    c = cr/Navg % 6;       /* red */
X                  if (c) c += 5; /* red */
X                  else c = 1;
X                 }
X                 else if (4*cb > 3*(cr+cg))
X                 {    c = cb/Navg % 6;       /* blue */
X                  if (c) c += 10; /* red */
X                  else c = 1;
X                 }
X                 else c = (cg/Navg+1) % 6;        /* white */
X                 break;
X             }
X             dprintf("%d,%d,%d -> %d\n", cr, cg, cb, c);
X             vsf_color(wHandle, c); /*draw colored box*/
X             v_bar(wHandle, pxy_array);
X            }
X            if ((ctr&15)== 0 && kbd() == -1)
X             goto redraw;
X         }
X         sprintf(str, "%s%03d:     RAYMOVI.PRG     rpp=%05.3f",
X              outname, fr, (float)ctr/((long)dx*dy));
X         di_header(1, str);
X    }   }
X    sprintf(str, "%s%03d.pi1", outname, fr);
X    f_write(str, colorInd-1);
X
X    movie(&bl, nob, step, Step, gravity, bounce, attraction);
X    write_ckpt("checkpt");
```

```
X    }
Xexit:
X    w_lClut(oldClut, (long)0);
X    w_close();
X}
X
Xkbd()
X{   char key, fname[20];
X    int found_first = 0, fr1, rval=0;
X
X    if (Cconis() == 0)
X         return(0);
X
X    key = Cconin();
X    while (1)
X    {    sprintf(fname, "%s%03d.pi1", outname, fr);
X         switch(key)
X         {   case ('?'): case ('\0'):          /* help */
X              printf("\033E\n\n       RayMovi\n\
X              \nw - write \"%s\" from screen,\
X              \nr - read \"%s\" to screen,\
X              \np - play movie \"%sXXX.pi1\
X              \ne - exit\n\n", fname, fname, outname);
X              key = Cconin();
X              printf("\033E");
X              rval = -1;
X              break;
X             case ('D'):
X              debug = 1-debug;
X              return (rval);
X             case ('w'):
X              f_write(fname, colorInd-1);
X              return (rval);
X             case ('r'):    case ('p'):
X              fr1 = 1;
X              while (key == 'p' || key == 'r')
X              {    sprintf(fname,"%s%03d.pi1", outname, fr1++);
X                   if (f_read(fname, (long)0) <0)
X                   {    if (found_first)
X                        {    fr1 = 1;
X                             found_first = 0;
X                   }    }
X                   else
X                        found_first = 1;
X                   if (Cconis() != 0 || key == 'r' && found_first)
X                        key = Cconin();
X              }
X              rval = -1;
X              break;
X             case ('e'): case('\003'):
X              w_lClut(oldClut, (long)0);
X              w_close();
X              exit(0);
```

```
X            default:
X             return(rval);
X}   }    }
X
Xscene_inz(inz_file)
Xchar *inz_file;
X{   int h, i, j, k; char str[200], *buf;
X    if ((fp=fopen(inz_file, "r"))==0)
X    {   printf("can't open file '%s'\n", inz_file);
X    return(0);
X    }
X    j=0, h=0;
X    while (1)
X    {   int linenum;
X        buf = str;
X    for (k=0; (i = getc(fp)) != '\n' && i != EOF && ++k<sizeof(str)-1;)
X        *buf++ = i;
X    *buf = '\0';
X    linenum++;
X    printf("%d: %s\n", linenum, str);
X    buf = str;
X    if (i == EOF)
X        break;
X    switch (buf[0])
X    {   case ('p'):        /* physical ball properties */
X         if (j >= Nball)
X              panic("**** ERROR: too many balls (%d >= %d)", j,Nball);
X         if (7 !=sscanf(buf+1," %f,%f,%f,%f,%f,%f,%f", &bl[j].s.cent.x,
X              &bl[j].v.x, &bl[j].s.cent.y, &bl[j].v.y,&bl[j].s.cent.z,
X              &bl[j].v.z, &bl[j].s.rad)) goto errors;
X         j++;
X         break;
X        case ('o'):        /* optical ball properties */
X         if (h >= Nball)
X              panic("**** ERROR: Too many balls (%d >= %d)", h,Nball);
X         if(8 !=sscanf(buf+1," %f,%f,%f,%f,%f,%f,%f,%f", &bl[h].ior,
X              &bl[h].rfr, &bl[h].rfl, &bl[h].dif, &bl[h].amb,
X              &bl[h].red, &bl[h].green, &bl[h].blue)) goto errors;
X            bl[h].ior *= bl[h].ior;
X         h++;
X         break;
X        case ('v'):        /* define viewpoint */
X         if (3 != sscanf(buf+1, " %f,%f,%f", &vp.x, &vp.y, &vp.z))
X              goto errors;
X         break;
X        case ('l'):  /* define light source (rad = how fuzzy shadows are) */
X         if (4 != sscanf(buf+1, " %f,%f,%f,%f", &ls.cent.x, &ls.cent.y,
X              &ls.cent.z, &ls.rad)) goto errors;
X         break;
X        case ('r'):  /* define number of rays per pixel and color scheme */
X         if (2 != sscanf(buf+1, " %f, %d", &rPp, &colorInd))
X              goto errors;
X         colorInd = colorInd<1? 1: colorInd>=3? 3: colorInd;
```

```
X         rPp = sqrt(rPp>16.0? 16.0: rPp);
X         break;
X        case ('g'):     /* gravity, etc */
X         if (3 != sscanf(buf+1, " %f, %f, %d", &gravity, &attraction,
X              &bounce)) goto errors;
X         gravity /= Step * 10;
X         break;
X        case ('f'):     /* define the filename */
X         while(*(++buf) == ' '); /* skip leading spaces */
X         for (i=0; i<5 && *buf!=','; i++)
X              outname[i] = *buf++;
X         outname[i] = 0;
X
X         if (2 != sscanf(buf+1, "%d,%d", &fr, &frames))
X              goto errors;
X         break;
X        case ('\t'):  /* comment */
X        case (' '):
X        case ('\0'):
X         break;
X        default:
X        errors:
X         printf("**** ERROR in line %d:\n%s\n", linenum, buf);
X    }   }
X    fclose(fp);
X    if (h != j)
X    panic("Number of 'o' lines (%ld) mismatches\n number of 'p' lines (%ld)\n",
X         (long)h, (long)j);
X    nob = j;
X    return(1);
X}
X
Xwrite_ckpt(name)
Xchar *name;
X{   int i;
X    if ((fp = fopen(name, "w")) != 0)
X    {
X        fprintf(fp, "f %s, %d, %d\n", outname, fr+1, frames);
X        for (i=0; i<nob; i++)
X        {    fprintf(fp, "p %f,%f,%f,%f,%f,%f,%f\n", bl[i].s.cent.x,
X             bl[i].v.x, bl[i].s.cent.y, bl[i].v.y, bl[i].s.cent.z,
X             bl[i].v.z, bl[i].s.rad);
X         fprintf(fp, "o %f,%f,%f,%f,%f,%f, %f,%f,%f\n\n",sqrt(bl[i].ior),
X             bl[i].rfr, bl[i].rfl, bl[i].dif, bl[i].amb,
X             bl[i].red, bl[i].green,bl[i].blue);
X        }
X        fprintf(fp, "v %f,%f,%f\n", vp.x, vp.y,vp.z);
X        fprintf(fp, "l %f,%f,%f,%f\n",
X         ls.cent.x, ls.cent.y, ls.cent.z, ls.rad);
X        fprintf(fp, "r %f, %d\n", rPp*rPp, colorInd);
X        fprintf(fp, "g %f, %f, %d\n", gravity*Step* 10, attraction, bounce);
X        fclose(fp);
X}   }
```

```
X
Xchar getline(str, sizeofstr)
Xchar *str;
Xint sizeofstr;
X{   char *buf, c;
X    int k, i;
X
X    buf = str;
X    for (k = sizeofstr; k > 0;)
X    {   c = i = Cconin();
X    if (c == '\n' || c == '\r' || c == ' ')
X        return (c);
X    else if (c == '\003' || c == '\214') /* ^c or Undo */
X        return ('\003');
X    else if (c == '\b')
X    {   if (k < sizeofstr)
X        {   *--buf = 0;
X            k++;
X        }else
X            printf(" ");
X    }
X    else
X    {   *buf++ = c;
X        *buf = '\0';
X        k--;
X}   }   }
X
Xpanic(s,l1,l2)
Xchar *s;
Xlong l1,l2;
X{   printf("\n");
X    printf(s,l1,l2);
X    printf("\npress any key to continue");
X    Cconin();
X
X    if (wHandle)
X    {   w_lClut(oldClut, (long)0);
X        w_close();
X    }
X    exit(1);
X    atoi();
X}
SHAR_EOF
chmod 0600 MAIN.C || echo "restore of MAIN.C fails"
sed 's/^X//' << 'SHAR_EOF' > MAKEFILE &&
XNAME1 = raymovi
XOBJS1 = gemstart.o main.o wndows.o file.o movie.o shade.o find.o support.o
X
XNAME2 = blastram
XOBJS2 = gemstart.o blastram.o wndows.o file.o
X
XNAME3 = bug
XOBJS3 = bug.o
```

```
X#---------------------------------------- invariant:
XT = g:
XBIN = c:\bin
X
XCPFLAGS = -i c:\include\ -DATARIST
XFpFlag = -f
XC0FLAGS =
XC1FLAGS =
XASFLAGS = -l -u -s $(BIN)\
X
X# -s d:\ -f d:\ ,,,   -i c:\bin\as68symb.dat
X
X.c.o:
X    @$(ECHO) *********     $*.C:     *********
X    @$(RM) $*.o
X    $(BIN)\$(CP68) $(CPFLAGS) $*.c $(T)\$*.i
X    $(BIN)\$(C068) $(T)\$*.i $(T)\$*.1 $(T)\$*.2 $(T)\$*.3 $(FpFlag) $(C0FLAGS)
X    @$(RM) $(T)\$*.i
X    $(BIN)\$(C168) $(T)\$*.1 $(T)\$*.2 $(T)\$*.s $(FpFlag) $(C1FLAGS)
X    @$(RM) $(T)\$*.1 $(T)\$*.2
X#   $(BIN)\$(AS) $(ASFLAGS) $(T)\$*.s
X#   @$(CP) $(T)\$*.o $*.o
X    $(BIN)\mac.prg -6 -o $*.o $(T)\$*.s
X    @$(RM) $(T)\$*.s       # $(T)\$*.o
X
X.s.o:
X    @$(ECHO) *********     $*.S:     *********
X#   $(BIN)\$(AS) $(ASFLAGS) $*.s
X    $(BIN)\mac.prg -6 -o $*.o $*.s
X
X$(NAME1).prg : $(OBJS1)
X    @$(ECHO)
X    $(BIN)\aln.prg -s -u -o $(NAME1).prg -c $(NAME1).lnk
X    @$(ECHO) * * * * * * * *  $(NAME1).ttp made  * * * * * * * *
X
X$(NAME2).ttp : $(OBJS2)
X    @$(ECHO)
X    $(BIN)\aln.prg -s -u -o $(NAME2).prg -c $(NAME2).lnk
X    @$(CP) $(NAME2).prg $(NAME2).ttp
X    @$(RM) $(NAME2).prg
X    @$(ECHO) * * * * * * * *  $(NAME).ttp made  * * * * * * * *
X
X$(NAME3).prg : $(OBJS3)
X    @$(ECHO)
X    $(BIN)\aln.prg -s -u -o $(NAME3).prg -c $(NAME3).lnk
X    @$(ECHO) * * * * * * * *  $(NAME3).ttp made  * * * * * * * *
X
SHAR_EOF
chmod 0600 MAKEFILE || echo "restore of MAKEFILE fails"
sed 's/^X//' << 'SHAR_EOF' > MOVIE.C &&
X/* movie.c        inertia, gravity, bouncing */
X
X#include "rtd.h"
```

```
X#include <math.h>
X
Xdouble dot();
Xfloat tx43;
X#define cube(a) ((tx43=a)*(tx43)*(tx43))
X#define density 1.0
X
Xmovie(bl, nob, step, Step, gravity, bounce, attraction)
Xstruct ball *bl;
Xint nob, step, Step, bounce;
Xfloat gravity, attraction;
X{   register int i, j;
X    register struct ball *bli, *blj;
X    struct vector f;
X    for (step = 0; step<Step; step++)
X    {    for (i=0, bli=bl; i<nob; i++, bli++)
X         {    if (bounce) for (j=i+1, blj=bli+1; j<nob; j++, blj++)
X              {    register float
X                        massi, massj, /* mass of balls (i&j) */
X                        fi, fj,     /* mass fraction (i&j) */
X                        tmp;
X                   struct vector cmv,  /* center mass velocity*/
X                          rvi, rvj,    /* relative vel (i&j)  */
X                          d;    /* distance between i&j*/
X                   sv(&d, &bli->s.cent, &blj->s.cent);
X                   vecl(&d);
X                   if (d.l <= bli->s.rad + blj->s.rad)
X                   {    tmp = bli->s.rad; massi = tmp*tmp*tmp;
X                        tmp = blj->s.rad; massj = tmp*tmp*tmp;
X                        fi = massi/(massi+massj);
X                        fj = massj/(massi+massj);
X                        cmv.x = fi*bli->v.x + fj*blj->v.x;
X                        cmv.y = fi*bli->v.y + fj*blj->v.y;
X                        cmv.z = fi*bli->v.z + fj*blj->v.z;
X
X                          /*    rvi = bli->v - cmv      */
X                        sv(&rvi, &bli->v, &cmv);
X                        tmp = dot(&d,&rvi);
X                          /*    d = 2*|d.rvi|/|d.d|*d   */

X                        scamult(2.0*tmp/(d.l*d.l),&d);
X                          /*    bli->v = d - rvi + cmv  */

X                        sv(&rvi, &rvi,&d);
X                        av(&bli->v, &cmv, &rvi);
X
X                        vecl(&d);
X                        sv(&rvj, &blj->v, &cmv);
X                        tmp = dot(&d,&rvj);
X                        scamult(2.0*tmp/(d.l*d.l),&d);
X                        sv(&rvj, &rvj,&d);
X                        av(&blj->v, &cmv, &rvj);
X              }    }
```

```
X             if (attraction != 0.0)
X             {    register float dl, tmp;
X                  struct vector d;
X                  f.x = f.y = f.z = 0.0;
X                  for (j=0, blj=bl; j<nob; j++, blj++) if (i != j)
X                  {
X                       sv(&d, &bli->s.cent, &blj->s.cent);
X                       dl = sqrt(d.x*d.x + d.y*d.y + d.z*d.z);
X                       tmp = blj->s.rad/dl; tmp=tmp*tmp*tmp;
X                       scamult(tmp, &d); /* mass/(r*r) */
X                       av(&f, &f, &d);
X                  }
X                  scamult(attraction/(density*Step), &f);
X                  sv(&bli->v,  &bli->v, &f);
X         }    }
X         for (i=0, bli=bl; i<nob; i++, bli++)
X         {    bli->s.cent.x += bli->v.x/Step;
X
X              bli->v.y -= gravity;
X              bli->s.cent.y += bli->v.y/Step;
X              if (bli->s.cent.y < bli->s.rad && bli->v.y < 0)
X                   bli->v.y = - bli->v.y;
X
X              bli->s.cent.z += bli->v.z/Step;
X              if (bli->s.cent.z < bli->s.rad && bli->v.z < 0)
X                   bli->v.z = - bli->v.z;
X}   }    }
SHAR_EOF
chmod 0600 MOVIE.C || echo "restore of MOVIE.C fails"
sed 's/^X//' << 'SHAR_EOF' > PEARLS &&
Xf pearl, 1, 24             PEARL001.PI1,...,PEARL024.PI1
Xr 1.0, 3          medium-rez (1.0 rays per pixel), 3 colors
Xg 32.0, 0.0, 1             gravity, no mutual attraction, bounce
X
X  (x     vx, y,     vy    z,     vz) radius
Xp 195.0,-20, 130.0,  0,   250.0, 10,30.0     these are 4 linked balls:
Xp 150.0, -5,      100.0, 20,   250.0, 10, 30.0     "
Xp 105.0,  5,     70.0, 20,   250.0,-20, 30.0  "
Xp 60.0,  20,     40.0,  0,   250.0,-20, 30.0  "
Xp  0.0, -10,    300.0, 20,   600.0,  0,300.0  this is the big ball
Xp 200.0, -8,     50.0, 35,    60.0, 30, 50.0  a medium size (lone) ball
X
X  ior  refr refl difu amb    red grn blue
Xo 0.0, 0.0, 0.3, 0.6, 0.1,   1.0,0.0,0.0   diffuse red, somewhat reflective
Xo 0.0, 0.0, 0.6, 0.3, 0.1,   1.0,1.0,1.0   reflective, somewhat white
Xo 0.0, 0.0, 0.9, 0.0, 0.1,   1.0,1.0,1.0   reflective, some ambient
Xo 0.0, 0.0, 0.0, 0.8, 0.2,   1.0,0.0,0.0   diffuse red, some ambient
Xo 0.0, 0.0, 0.7, 0.1, 0.2,   1.0,1.0,1.0   reflective and ambient (the big one)
Xo 1.6, 0.8, 0.0, 0.0, 0.2,   1.0,1.0,1.0   refractive
X
Xv 215.0,     100.0,       -550.0         viewpoint
Xl 150.0,     950.0,          0.0,    100.0    light source
SHAR_EOF
```

```
chmod 0600 PEARLS || echo "restore of PEARLS fails"
sed 's/^X//' << 'SHAR_EOF' > RAYMOVI.DOK &&
X             RAYMOVI.PRG
X
X   RAYMOVI.PRG renders 3-dimensional scenes in a world of reflective and
Xrefractive spheres, using ray tracing techniques. Movies formed from
Xsequences of frames can be generated, showing reflective balls linked and
Xbouncing in a gravitational field. They resemble a string of bouncing pearls.
X
X   The amount of computation required to produce one ray traced image is
Xenormous -- on an ST it is measured in hours. To get animated effects,
Ximages must be precomputed in (overnight) batch runs. RAYMOVI provides the
Xability to save sequences of images on disk in Degas PI1 format, for later
Xreplay by BLASTRAM.TTP (also included).
X
X   RAYMOVI.PRG generates a low resolution approximation of each picture first,
Xfollowed by refinement to successively higher resolutions. In this way,
Xa rough image is seen quickly for (somewhat) interactive debugging. Scenes
Xcan be changed without recompilation of RAYMOVI.PRG -- the size, position,
Xvelocity and optical properties of the balls in the scenes are specified
Xnumerically in the initialization files.
X
X   The ray tracing heart of RAYMOVI was taken from net.sources, documented
XFredrich Knauss of University of California, San Diego. I've ported it to
Xthe ST, added animation, (primitive) color, and the various other features
Xyou see here. I place RAYMOVI in the public domain, for non-comercial use only.
X
X   RAYMOVI.PRG runs only on a color ST in low resolution. 1 Meg is required
Xto generate images with resolutions greater than one ray per pixel or for
Xreplay of movies from RAM. Operation with only 512K is possible, but will
Xproduce slightly aliased pictures (one ray per pixel) which can only be
Xdisplayed statically, slowly from floppy, or from hard disk (if you have
Xa disk blaster program).
X
X   The rest of this document contains sections entitled Getting Started,
XConfiguration File Format, and Checkpointing. The sources are available
Xupon request. With enough interest I could post them.
X
X
X
X             GETTING STARTED
X
X   The screen must be set to low resolution mode before running RAYMOVI.PRG.
XOnce started, '?' will give help. Keys 'r' reads and 'w' writes the current
Xscreen using the current disk file, whose name is displayed in the upper
Xleft corner of the screen. Key 'e' exits. The average number of rays shot
Xthrough each pixel is also displayed in the margin and updated as the
Xcomputation develops.
X
X   After unpacking RAYMOVI, the first logical thing to do is to start it up
Xwith the configuration file PEARLS.  After a few minutes, you'll see
Xa rough image, with the resolution increasing as you watch. You might
Xwait an hour or so to get complete development, but if you get bored, hit
```

```
Xthe 'r' key. This will read the one pre-computed picture PEARL001.PI1 which
Xcame in the "arc".
X
X   Now put RAYMOVI aside until you are willing to dedicate a large chunk of
XST time to make a movie. When you have 10-20 hours free, start RAYMOVI,
Xagain with PEARLS. (Before you start, be sure you have a few hundred K
Xfree on the disk for the images which will be produced.) To view the movie
Xproduced, exit ('e') and start BLASTRAM.TTP, supplying the argument
XPEARL in the dialog box. With BLASTRAM running, type '?' for help.
XCheckpointing may be used if you wish to add more frames to the movie.
X
X
X
X             CHECKPOINTING
X
X
X   Since RAYMOVI.PRG runs are so long, checkpointing is important. (Actually
XI put it in because my sons rebooted the ST at 3:00 every day after school!).
SHAR_EOF
echo "End of part 1, continue with part 2"
echo "2" > s2_seq_.tmp
exit 0


From -7064759667205603078
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,14e20c611314b2e1,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-12-19 13:58:49 PST
Path: utzoo!utgpu!watmath!clyde!att!osu-cis!tut.cis.ohio-state.edu!rutgers!ucsd!
nosc!helios.ee.lbl.gov!pasteur!agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i079: raymovi2 -- Raytrace movie generator  part02/02
Keywords: shar, Alcyon, low
Message-ID: <5800@saturn.ucsc.edu>
Date: 19 Dec 88 21:58:49 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 980
Approved: koreth@ssyx.ucsc.edu

Submitted-by: aking@bbn.com (Allen King)
Posting-number: Volume 1, Issue 79
Archive-name: raymovi2/part02

#!/bin/sh
# this is part 2 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file RAYMOVI.DOK continued
#
CurArch=2
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
```

```
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> RAYMOVI.DOK
XThus following the completion of every frame, RAYMOVI saves the current
Xconfiguration on the disk so that if the computation of the next frame
Xis interupted, it may be restarted.  The checkpointing information
Xis written in a file named CHECKPT, in configuration file format. Starting
XRAYMOVI with CHECKPT will continue an interupted computation, redoing
Xonly any work which was in progress on the interupted frame.
X
X
X
X             CONFIGURATION FILE FORMAT
X
X
X   The following section discusses the format of the configuration files,
Xwhich specify initial ball positions, velocities and optical properties,
Xas well as lighting and viewing perspectives, picture resolution, and gravity.
XThe configuration file name to be used is specified initially with a dialog.
XThe file is then read line at a time. The character in column 1 specifies the
Xmeaning of the rest of each line. Lines may be up to 200 characters long. Lines
Xbeginning with whitespace are comments. The following other formats are
Xsupported:
X
X
X
X     - - - 'p' Format: - - -
X
Xspecifies the physical size and velocity of a ball. There are 7 properties
Xrequired for each ball, and they all must be specified on one line in the
Xfollowing order:
X
X (1) x coordinate of center         (2) x velocity
X (3) y coordinate          (4) y velocity
X (5) z coordinate          (6) z velocity
X (7) radius
X
XSome Notes:
X A) In the coordinate system used, Y is upward, X is to the right, and Z
X    is depth (increasing Z moves farthur away).
X B) Balls will bounce off the floor (the plane Y == 0) and an invisible
X    viewing port (the plane Z==0). Be sure to start them so that the whole
X    extent of the ball is above the floor and farther away than the
X    invisible viewing port.
X C) If balls start out overlapping, they will stick together.
X    This turned out to be a serendipitous bug in the bounce algorithm,
X    and produces the string of pearls from which PEARLS gets its name.
X D) The current version is compiled for up to 15 balls maximum.
```

```
X
X   The first p-format statement in the file defines the "first" ball RAYMOVI
Xknows about, the second the second and so on. Here are 6 balls which
Xform an interesting scene:
X
X   x     vx      y     vy    z      vz   radius
X  (1)    (2) (3)    (4)   (5)    (6)  (7)       (see (1)-(7) above)
Xp 195.0,-20, 130.0, 0,    250.0, 10,  30.0    these are the 4 linked balls:
Xp 150.0, -5,      100.0, 20,   250.0, 10,  30.0        "
Xp 105.0,  5,  70.0, 20,   250.0,-20,  30.0         "
Xp  60.0, 20,  40.0,  0,   250.0,-20,  30.0         "
Xp   0.0,-10, 300.0, 20,   600.0,  0, 300.0    this is the big ball
Xp 200.0,  2,  50.0,  5,    60.0, 30,  50.0    a medium size (lone) ball
X
X
X
X
X     - - - 'o' Format: - - -
X
Xspecifies the optical properties of a ball. The following 8 properties
Xmust be put on one line, and in order:
X
X (1) index of refraction
X
X (2) amount of refraction (transparent)
X (3) amount of reflection (mirror)
X (4) amount of diffuse light (omnidirectional)
X (5) amount of ambient light (from light source)
X
X (6) surface color redness
X (7) surface color greenness
X (8) surface color blueness
X
X
XSome notes are in order to make realistic objects:
X A) All parameters should be positive or zero.
X B) Items (2) through (5) should sum to less than 1.0
X C) Surface colors (6) through (8) modulate the effect the diffuse and ambient
X    reflective properties defined by (4) and (5). None should be greater than
X    1.0.
X C) Due to the size of the ST's color palette, surface color items (6)
X    through (8) are constrained to be patterns corresponding to one of the
X    following three colors:
X
X        white:        (6) == (7) == (8) == 1.0
X        red:     (6) == (7) == 1.0, (8) == 0.0
X        blue:         (6) == 0.0, (7) == (8) == 1.0
X
X
X   The i'th 'o' Format line refers to the same ball as the i'th 'p' Format line.
XHere are the optical properties for the scene:
X
X  (1)  (2)  (3)  (4)  (5)    (6) (7) (8)
```

```
Xo 0.0, 0.0, 0.3, 0.6, 0.1,   1.0,0.0,0.0   diffuse red, somewhat reflective
Xo 0.0, 0.0, 0.6, 0.3, 0.1,   1.0,1.0,1.0   reflective, somewhat white
Xo 0.0, 0.0, 0.9, 0.0, 0.1,   1.0,1.0,1.0   reflective, some ambient
Xo 0.0, 0.0, 0.0, 0.8, 0.2,   1.0,0.0,0.0   diffuse red, some ambient
Xo 0.0, 0.0, 0.7, 0.1, 0.2,   1.0,1.0,1.0   reflective and ambient (the big one)
Xo 1.6, 0.8, 0.0, 0.0, 0.2,   1.0,1.0,1.0   refractive
X
X
X
X
X     - - - 'v' Format: - - -
X
Xspecifies the viewpoint of the display. x, y, and z values are required.
X
Xv 215.0,     100.0,       -550.0
X  |     |         z coordinate of viewpoint
X  |     y coordinate of viewpoint
X  x coordinate of viewpoint
X
XAll rays start at the viewpoint and pass through a rectangle in the Z=0
Xplane. The rectangle is sized according to the size of the screen -- namely
Xon a 320x200 low resolution screen, it is defined by 0.0 < X < 320.0 and
X0.0 < Y < 200.0. (Actually, borders eat a few pixels on all sides.)
X
X
X     - - - 'l' Format: - - -
X
X   specifies the location of the lightsource, and its diameter. Like before
Xits x, y, and z coordinates, now followed by the diameter of the light.
X
Xl 150.0,     950.0,          0.0,        100.0
X
X
X     - - - 'r' Format: - - -
X
Xspecifies the resolution of the picture to be produced, in Rays projected
Xthrough each Pixel, followed by the color schema index, a number from 1
Xthrough 3.
X
Xr .20, 3
X
X   The inputted value of Rays Per Pixel will be rounded up to the next
Xgreater power of 4.0. Thus in the line above, the .20 will be rounded up
Xto (4.0)**(-1) = .25.
X
X   The time to compute images varies linearly with the resolution (Rays Per
XPixel) chosen. The constant of proportionality varies greatly depending on
Xthe complication of the scene, but for the 6-ball scene here on a 520ST:
X
X      .0625 rays per pixel    4 min
X      .25 rays per pixel     15 min
X     1.0 rays per pixel       1 hr
X     4.0 rays per pixel       4 hrs
```

X   16.0 rays per pixel   16 hrs
X
XI tend to use .25 rays per pixel to experiment with positions and velocities,
Xsince one can generate a couple dozen frames overnight. Once the positions
Xlook good, a few nights of 4.0 rays per pixel produces good looking images.
XI can tell the difference between 16.0 and 4.0 rays per pixel (sort of) but
Xit is hardly worth the time it takes. 0.625 is skimping, but its quick.
X
X
X   The color index specifies how the 16 colors of the low resolution monitor
Xare divided amongst hues and saturation. Three values are accepted currently:
X
X   1:   white   16 saturations
X   2:   white, red   8 saturations each
X   3:   white, red, blue   6 saturations each
X
XAlthough the ray tracing algorithm computes arbitrary red/green/blue hues,
Xthe palate limitations of the ST prohibit their display.
X
X
X
X
X   - - - 'g' Format: - - -
X
Xspecifies the strength of gravity, the mutual attraction between balls,
Xand if balls bounce (in order). The following is realistic:
X
Xg 32.0, 0.3, 1
X
X(Note: if balls do not bounce, they may occupy the same space. In that case
Xthe current the ray tracing algorithm does not properly render refractive
Xobjects.)
X
X   - - - 'f' Format: - - -
X
Xspecifies the first 5 characters of the filename to be used to save
Xcompleted pictures. (The rest of the name is the frame sequence number,
Xand the suffix .PI1) Also specified are the starting and ending frame
Xnumbers to be computed (positive integers less than 999).
X
Xf pearl, 1, 20
X
Xgenerates 20 frames on the disk, labeled PEARLL001.PI1 through PEARL020.PI1.
XAt 32K per frame, that will just about fill a double sided drive.
X
X
X
X   --------------
X
X   Thats it for now. Tell me of you ventures, send me interesting
Xconfigurations you find, and enjoy!
X
X

```
X
X         Allen King    aking  bfly-vax.bbn.com  ucbvax
X         30 Gibson St  1-617-449-3359 evenings
X         Needham Ma.
X         02192
SHAR_EOF
chmod 0600 RAYMOVI.DOK || echo "restore of RAYMOVI.DOK fails"
sed 's/^X//' << 'SHAR_EOF' > RAYMOVI.LNK &&
X\include\gemstart.o main.o wndows.o movie.o file.o shade.o find.o support.o
X\include\aesbind \include\vdibind \include\gemlib \include\osbind.o
X\include\libf
SHAR_EOF
chmod 0600 RAYMOVI.LNK || echo "restore of RAYMOVI.LNK fails"
sed 's/^X//' << 'SHAR_EOF' > README &&
X             RAYMOVI.PRG
X
X   RAYMOVI.PRG renders 3-dimensional scenes in a world of reflective and
Xrefractive spheres, using ray tracing techniques. Animated sequences of
Xframes can be generated, showing reflective balls linked and bouncing in
Xa gravitational field. They resemble a string of bouncing pearls.
X
X   RAYMOVI.PRG runs only on a color ST in low resolution. 1 Meg is required
Xto generate images with resolutions greater than one ray per pixel or for
Xreplay of movies from RAM. Operation with only 512K is possible, but will
Xproduce slightly aliased pictures (one ray per pixel) which can only be
Xdisplayed statically, slowly from floppy, or from hard disk (if you have
Xa disk blaster program).
X
X        ---------------------------------------------------
X
X   The files in RAYMOV2S.ARC, the source arc, are:
X
XC Sources:   BLASTRAM.C, MAIN.C, WNDOWS.C, FILE.C, MOVIE.C, SHADE.C,
X         FIND.C, SUPPORT.C
XInclude files:    RTD.H, STYLE.H, COLORS.H, EXTERN.H
XPgm Generation: MAKEFILE, BLASTRAM.LNK, RAYMOVI.LNK
XRAYMOVI.DOK  documentation
XPEARLS       the numeric description of an interesting scene
X
X   RAYMOVI was last compiled with ALCYON. The MAKEFILE and *.LNK are
X(therefore) pretty crufty. Use them accordingly.
X
X   RAYMOVI requires an 8K stack. Modify your copy of gemstart.s (or
Xwhatever defines your stack size) appropriately.
X
X
X    Allen King    aking  pebbles.bbn.com  ucbvax
X    30 Gibson St  1-617-449-3359 evenings
X    Needham Ma.
X    02192
X
X
X
```

```
SHAR_EOF
chmod 0600 README || echo "restore of README fails"
sed 's/^X//' << 'SHAR_EOF' > RTD.H &&
X#define NCOLORS 3
X#define HUGE 1e15
X#define EPSILON 1e-3
X
Xstruct vector
X    {
X    double x;
X    double y;
X    double z;
X    double l;
X    double xzl;
X    } ;
X
Xstruct ray
X    {
X    struct vector org;
X    struct vector dir;
X    } ;
X
Xstruct sphere
X    {
X    struct vector cent;
X    double rad;
X    } ;
X
Xstruct ball
X    {
X    struct sphere s;
X    double ior;
X    double rfr;
X    double rfl;
X    double dif;
X    double amb;
X    double red;
X    double green;
X    double blue;
X    struct vector v;
X    };
X
Xstruct mat
X    {
X    struct vector x;  /* first !row! */
X    struct vector y;  /*second !row! */
X    struct vector z;   /* third !row! */
X    };
X
SHAR_EOF
chmod 0600 RTD.H || echo "restore of RTD.H fails"
sed 's/^X//' << 'SHAR_EOF' > SHADE.C &&
X/*
```

```
XFrom harvard!husc6!caip!nike!ucbcad!ucbvax!sdcsvax!net1!fritzz 25 Jul 86 21:35:59
GMT
XArticle 2915 of net.sources:
XRelay-Version: version B 2.10.2 9/18/84; site bbnccv.UUCP
XPath: bbnccv!harvard!husc6!caip!nike!ucbcad!ucbvax!sdcsvax!net1!fritzz
X>From: fritzz@net1.UCSD.EDU (Friedrich Knauss)
XNewsgroups: net.sources
XSubject: basic ray tracing: part 1 (tracer.c)
XMessage-ID: <190@net1.UCSD.EDU>
XDate: 25 Jul 86 21:35:59 GMT
XDate-Received: 27 Jul 86 01:07:29 GMT
XReply-To: fritzz@net1.UUCP (Friedrich Knauss)
XOrganization: University of California, San Diego
X*/
X/*
X * this subroutine does all the gritty work- it calculates
X * what shade each pixel should be. I like recursion.
X */
X
X#include <math.h>
X#include "rtd.h"
X#include "extern.h"
X#define time(x) 0
X
X/* LEVEL defines number of levels of recursion */
X#define LEVEL 5
Xint debug;
X#define dprintf if (debug) printf
X
Xshade (r, red, green, blue)
Xstruct ray *r;
Xdouble *red, *green, *blue;
X{   int     i, c,
X           refract ();
X    struct ray  refr;
X    double  rd, gn, blu, tmp;
X    double  lght_red, lght_green, lght_blue,
X           x, y, z,
X           l, k,
X           dot (), find (), findo ();
X    struct vector   new, norm;
X    struct mat  trans;
X    struct sphere   ss;
X
X    if (++level <= LEVEL)
X    {   c = -1;
X    l = HUGE;
X
X    dprintf("LEVEL %d: ", level);
X        /* get vector length and xz component for mt() */
X    vecl (&r->dir);
X    vexzl (&r->dir);
X
```

```
X        /* make a transform matrix that rotates something in space so
X            that the ray will be aligned with the x axis */
X
X    mt (&r->dir, &trans);
X
X        /* for starters we find out whether we hit anything. */
X
X    for (i = 0; i < nob; i++)
X    {   ss.rad = bl[i].s.rad;
X        sv (&ss.cent, &bl[i].s.cent, &r->org);
X        if ((k = find (&trans, &ss)) > 0.0 && k < l)
X        {    c = i;
X         l = k;
X        }
X    }
X    if (c >= 0.0)           /* WE HIT SOMETHING */
X    {   x = l * trans.x.x;
X        y = l * trans.x.y;
X        z = l * trans.x.z;
X        mv (x, y, z, &new);
X
X        /* move the new orgin of the ray to the intersection */
X
X        av (&refr.org, &new, &r->org);
X        av (&r->org, &new, &r->org);
X        mv (r->dir.x, r->dir.y, r->dir.z, &refr.dir);
X
X        /* get a normal vector for the intersection point */
X
X        sv (&norm, &r->org, &bl[c].s.cent);
X        vecl (&norm);
X
X        /* ambient lighting */
X
X        tmp = 255.0 * bl[c].amb;
X        lght_red = tmp * bl[c].red;
X        lght_green = tmp * bl[c].green;
X        lght_blue = tmp * bl[c].blue;
X        dprintf(" ball[%d]", c);
X        dprintf(", ambient=%d,%d,%d\n",
X         (int)lght_red, (int)lght_green, (int)lght_blue);
X
X        /* shaded lighting (diffuse). subroutine shadow is in find.c */
X
X        if (bl[c].dif != 0.0)
X        {    sv (&new, &ls.cent, &r->org);
X         vecl (&new);
X         if ((k = dot (&new, &norm)) > 0.0)
X         {   shadow (&r->org, &rd, &gn, &blu);
X             dprintf("diffuse = %d, %d, %d\n",(int)rd,(int)gn,(int)blu);
X             tmp = bl[c].dif * k / (new.l * norm.l);
X             lght_red   += tmp * rd * bl[c].red;
X             lght_green += tmp * rd * bl[c].green;
```

```
X            lght_blue  += tmp * rd * bl[c].blue;
X        }   }
X
X        /*reflection... easy */
X
X        if (bl[c].rfl != 0.0)
X        {    vecl (&norm);
X         /* make the normal unit length */
X         scamult ((1.0 / norm.l), &norm);
X         /* get the length of the ray's component in the normal direction */
X         x = 2.0 * dot (&norm, &r->dir);
X         scamult (x, &norm);
X         /* subtract double the normal component- !reflection! */
X         sv (&r->dir, &r->dir, &norm);
X         shade (r, &rd, &gn, &blu);
X         dprintf("reflection = %d, %d, %d, tmp=%d\n",
X              (int)rd, (int)gn, (int)blu, (int)tmp);
X         tmp = bl[c].rfl;
X         lght_red   += tmp * rd;
X         lght_green += tmp * gn;
X         lght_blue  += tmp * blu;
X        }
X
X        /* refraction. this is ugly, which is why I choose to deal with
X           it in it's own subroutine which comes after this one */
X
X        if (bl[c].rfr != 0.0)
X        {    dprintf(" refraction\n");
X         refract (&refr, &bl[c], &rd, &gn, &blu);
X         tmp = bl[c].rfr;
X         dprintf("refraction = %d, %d, %d\n",(int)rd,(int)gn,(int)blu);
X         lght_red   += tmp * rd;
X         lght_green += tmp * gn;
X         lght_blue  += tmp * blu;
X    }   }
X    else              /* hit no objects... */
X    {   if ((r->dir.y) < 0.0) /* crosses floor */
X        {    z = -(r->org.y) / (r->dir.y);
X         (r->org.x) += z * (r->dir.x);
X         (r->org.z) += z * (r->dir.z);
X         (r->org.y) = 0.0;
X         dprintf("floor (%d, %d): ", (int)r->org.x, (int)r->org.z);
X         if (  ((int) ((r->org.x) / 9.0) % 8 == 0)
X            || ((int) ((r->org.z) / 9.0) % 8 == 0))
X         {   lght_red = lght_green = lght_blue = 0.0;
X             dprintf("black. 0,0,0\n");
X         }        /* this is for texture, grid on the ground */
X         else          /* get shading for the squares... */
X         {   sv (&new, &ls.cent, &r->org);
X             vecl (&new);
X             shadow (&r->org, &rd, &gn, &blu);
X             tmp = 0.6 * new.y / new.l;
X             lght_blue   = tmp * blu + 100.0;  /* tile pattern: */
```

```
X            if ((int)r->org.x/72 & (int)r->org.z/72 & 0x15)
X            {    lght_red = lght_green = 0.0;         /* blue */
X                dprintf("blue. %d, %d, %d\n",
X                 (int)lght_red, (int)lght_green, (int)lght_blue);
X            }
X            else
X            {    lght_red   = tmp * rd + 100.0;       /* white */
X             lght_green = tmp * gn + 100.0;
X             dprintf("white. %d,%d,%d\n",
X                 (int)lght_red, (int)lght_green, (int)lght_blue);
X        }    }   }
X        else  /* didn't hit ground... sky */
X        {    lght_red = lght_green = lght_blue = 0.0;
X         dprintf("sky. %d,%d,%d\n",
X              (int)lght_red, (int)lght_green, (int)lght_blue);
X        }
X        dprintf("lght_= %d,%d,%d\n",
X         (int)lght_red, (int)lght_green, (int)lght_blue);
X    }   }
X    /* to many levels return 0 cause it shouldn't matter */
X    else
X    {   lght_red = lght_green = lght_blue = 0.0;
X    dprintf(">5 levels. 0, 0, 0\n");
X    }
X    level--;
X
X    if (lght_red < 0.0)    lght_red = 0.0;
X    if (lght_red > 255.0)  lght_red = 255.0;
X
X    if (lght_green < 0.0)  lght_green = 0.0;
X    if (lght_green > 255.0)     lght_green = 255.0;
X
X    if (lght_blue < 0.0)   lght_blue = 0.0;
X    if (lght_blue > 255.0) lght_blue = 255.0;
X
X    *red = lght_red;
X    *green = lght_green;
X    *blue = lght_blue;
X
X    dprintf("returned: %d, %d, %d\n", (int)*red, (int)*green, (int)*blue);
X}
X
Xint     refract (r, bll, red, green, blue)
Xstruct ray *r;
Xstruct ball *bll;
Xdouble *red, *green, *blue;
X{
X    struct vector   new, norm;
X    struct mat  trans;
X    double  l,
X            refk ();
X    struct sphere   ss;
X
```

```
X    sv (&norm, &r->org, &bll->s.cent);
X    vecl (&norm);
X    vecl (&r->dir);
X
X    /* get the addition factor for the normal */
X
X    scamult (refk (&norm, &r->dir, bll->ior), &norm);
X    av (&r->dir, &r->dir, &norm);
X
X    /* find the point where the ray leaves the sphere. just like shade. */
X
X    vexzl (&r->dir);
X    vecl (&r->dir);
X    mt (&r->dir, &trans);
X    ss.rad = bll->s.rad;
X    sv (&ss.cent, &bll->s.cent, &r->org);
X    l = findo (&trans, &ss);
X    mv (l * trans.x.x, l * trans.x.y, l * trans.x.z, &new);
X    av (&r->org, &r->org, &new);
X
X    /* redirect the ray and continue tracing */
X
X    sv (&norm, &r->org, &bll->s.cent);
X    vecl (&norm);
X    vecl (&r->dir);
X    scamult (refk (&norm, &r->dir, 1.0 / bll->ior), &norm);
X    av (&r->dir, &r->dir, &norm);
X
X    shade (r, red, green, blue);
X}
X

X
Xdouble  refk (nrm, in, ior)
Xstruct vector  *nrm, *in;
Xdouble  ior;
X{   double  dt, ln, li, ret;
X
X    dt = dot (nrm, in);
X    ln = nrm->x * nrm->x + nrm->y * nrm->y + nrm->z * nrm->z;
X    li = in->x * in->x + in->y * in->y + in->z * in->z;
X    ret = dt * dt - ln * li * (1 - ior);
X    if (ret < 0.0)
X    ret = 0.0;
X    if (dt < 0)
X    ret = (-dt - sqrt (ret)) / ln;
X    else
X    ret = (-dt + sqrt (ret)) / ln;
X    return (ret);
X}
X
Xprintv(s, v)
Xchar *s;
```

```
Xstruct vector *v;
X{   printf("%s.x = %f, .y = %f, .z = %f\n", s, v->x, v->y, v->z);
X}
SHAR_EOF
chmod 0600 SHADE.C || echo "restore of SHADE.C fails"
sed 's/^X//' << 'SHAR_EOF' > STYLE.H &&
X/* cosmetic C definitions */
X
X#define EQ ==
X#define IS ==
X#define NEQ !=
X#define ISNT !=
X#define AND &&
X#define OR ||
X#define NOT !
X#define NULL 0
X#define TRUE 1
X#define FALSE 0
X#define TRUE_STRING(i) (i IS TRUE ? "TRUE" : "FALSE")
X
X#define u unsigned
X
Xtypedef char boolean;
Xtypedef char *string;
X
X#define plural(i)       ( ((i) EQ 1) ? "":"s")
X
SHAR_EOF
chmod 0600 STYLE.H || echo "restore of STYLE.H fails"
sed 's/^X//' << 'SHAR_EOF' > SUPPORT.C &&
X/*
X *   supportive subroutines...
X */
X#include <math.h>
X#include <stdio.h>
X#undef min
X#include "rtd.h"
X#include "extern.h"
X#define time(x) 0
X
X/* returns dot product of two vectors */
X
Xdouble  dot (v1, v2)
X    struct vector  *v1,
X                   *v2;
X{
X    double  r;
X    r = v1->x * v2->x + v1->y * v2->y + v1->z * v2->z;
X    return (r);
X}
X
X
X/* multiplies a vector by a scalar */
```

```
X
Xscamult (scal, vect)
X    double  scal;
X    struct vector  *vect;
X{
X    vect->x *= scal;
X    vect->y *= scal;
X    vect->z *= scal;
X    vect->l *= scal;
X    vect->xzl *= scal;
X}
X
X
X/* loads a vector with given x,y,z values */
X
Xmv (x, y, z, _vp)
X    double  x,
X            y,
X            z;
X    struct vector  *_vp;
X{
X    _vp->x = x;
X    _vp->y = y;
X    _vp->z = z;
X}
X
X

X/* vector subtraction: v1 = v2 - v3 */
X
Xsv (v1, v2, v3)
X    struct vector  *v1,
X                   *v2,
X                   *v3;
X{
X    v1->x = v2->x - v3->x;
X    v1->y = v2->y - v3->y;
X    v1->z = v2->z - v3->z;
X}
X
X
X
X/* vector addition: v1 = v2 + v3 */
X
Xav (v1, v2, v3)
X    struct vector  *v1,
X                   *v2,
X                   *v3;
X{
X    v1->x = v2->x + v3->x;
X    v1->y = v2->y + v3->y;
X    v1->z = v2->z + v3->z;
X}
```

```
X
X
X/*  loads a transform matrix so that when vec is multiplied by it,
X    the result is (1,0,0). it does so with out deforming space */
X
Xmt (vec, trans)
X    struct vector  *vec;
X    struct mat *trans;
X{
X    vecl (vec);
X    vexzl (vec);
X
X    if (vec->xzl == 0.0) {
X    trans->x.x = 0.0;
X    trans->x.y = 1.0;
X    trans->x.z = 0.0;
X    trans->y.x = -1.0;
X    trans->y.y = 0.0;
X    trans->y.z = 0.0;
X    trans->z.x = 0.0;
X    trans->z.y = 0.0;
X    trans->z.z = 1.0;
X    }
X    else {
X    trans->x.x = (vec->x) / (vec->l);
X    trans->x.y = (vec->y) / (vec->l);
X    trans->x.z = (vec->z) / (vec->l);
X    trans->y.x = -(vec->x) * (vec->y) / ((vec->l) * (vec->xzl));
X    trans->y.y = (vec->xzl) / (vec->l);
X    trans->y.z = -(vec->z) * (vec->y) / ((vec->l) * (vec->xzl));
X    trans->z.x = -(vec->z) / (vec->xzl);
X    trans->z.y = 0;
X    trans->z.z = (vec->x) / (vec->xzl);
X    }
X}
X

X/* multiply vector by matrix: vc1 = m*vc2 */
X
Xmt_vec (vc1, m, vc2)
X    struct vector  *vc1,
X                   *vc2;
X    struct mat *m;
X{
X    double  x,
X            y,
X            z;
X
X    x = dot (&(m->x), vc2);
X    y = dot (&(m->y), vc2);
X    z = dot (&(m->z), vc2);
X    mv (x, y, z, vc1);
X}
```

```
X
X
X

X/* will update a file containing information on
Xhow we are coming along... useful for long runs*/
X
Xupdate (x, xmin, xmax)
X    float   x,
X            xmin,
X            xmax;
X{
X    FILE * fff;
X    static int  t;
X    static double   size,
X                    xm;
X    int     tt,
X            ttt,
X            hr,
X            min,
X            sec;
X
X
X    fprintf (stderr, "%d\n", (int) x);
X    return;
X
X
X    if (x == xmin) {
X/*    fff = fopen ("time.update", "w"); */
X    fff = stderr;
X    fprintf (fff, "starting...\n");
X/*    fclose (fff); */
X    t = time (0);
X    xm = xmin;
X    size = (xmax - xmin);
X    }
X    else {
X    x -= xm;
X    tt = time (0) - t;
X    ttt = (double) tt * size / x;
X    tt = (double) tt * (size - x) / x;
X    hr = tt / 3600;
X    min = (tt % 3600) / 60;
X    sec = tt % 60;
X/*    fff = fopen ("time.update", "w"); */
X    fff = stderr;
X    fprintf (fff, "%5.1f%% done.\n", 100.0 * x / size);
X    fprintf (fff, "%d:%2d:%2d left.\n", hr, min, sec);
X    hr = ttt / 3600;
X    min = (ttt % 3600) / 60;
X    sec = ttt % 60;
X    fprintf (fff, "%d:%2d:%2d  total estimate.\n", hr, min, sec);
X/*    fclose (fff); */
```

```
X    }
X}
X

X
X/* loads a vectors length... saves time to do this only when needed */
X
Xvecl (v)
X    struct vector *v;
X{
X    v->l = sqrt ((v->x) * (v->x) + (v->y) * (v->y) + (v->z) * (v->z));
X}
X
X
X/* loads a vectors xz component. needed for mt */
X
Xvexzl (v)
X    struct vector *v;
X{
X    v->xzl = sqrt ((v->x) * (v->x) + (v->z) * (v->z));
X}
X
X
SHAR_EOF
chmod 0600 SUPPORT.C || echo "restore of SUPPORT.C fails"
sed 's/^X//' << 'SHAR_EOF' > WNDOWS.C &&
X/* wndows.c
X *   manage the interface to GEM windows
X
XEntry Points:
X    w_open(type,name)           ***
X    w_close()
X        w_lClut(new, old)
X
XOutward Calls:
X    wOnewSize()
X
X */
X
X#include <obdefs.h>
X#include <define.h>
X#include <gemdefs.h>
X#include <vdibind.h>
X#include <osbind.h>
X
X#define MIN_WIDTH  (2*gl_wbox)
X#define MIN_HEIGHT (3*gl_hbox)
Xint gl_hchar, gl_wchar, gl_wbox, gl_hbox;     /* system sizes */
X
Xint ps_handle;    /* physical screen (workstation) handle */
Xint vs_handle;    /* virtual screen (workstation) handle */
Xint wi_handle;    /* window handle */
X
```

```
Xint xdesk,ydesk,hdesk,wdesk;    /* size of whole desktop */
Xint xwork,ywork,hwork,wwork;    /* size of current working window */
Xint xold,yold,hold,wold;        /* size of screen before FULL cmd */
X
Xint contrl[12], intin[128], ptsin[128], intout[128];
Xint ptsout[128];  /* storage for gem bindings */
X
Xchar *p_menu;
Xint WiKind;
X
Xw_open(kind, name, menue_id)
Xint kind;         /* properties of window (e.g. MOVER|CLOSER|...) */
Xchar *name;
Xint menue_id;          /* open menue? */
X{   char n2[30];
X    appl_init();
X    open_vwork();
X    open_window(kind, name);
X    WiKind = kind;
X    sprintf(n2, "%s.rsc", name);
X    open_rsc(n2, menue_id);
X    graf_mouse(ARROW,0x0L);
X    return (wi_handle);
X}
X
Xw_close()
X{   if (p_menu)
X        menu_bar(p_menu, FALSE);
X    wind_close(wi_handle);
X    graf_shrinkbox(xwork+wwork/2,ywork+hwork/
2,gl_wbox,gl_hbox,xwork,ywork,wwork,hwork);
X    wind_delete(wi_handle);
X    v_clsvwk(vs_handle);
X    appl_exit();
X}
X
Xint work_in[11];  /* Input to GSX parameter array */
Xint work_out[57]; /* Output from GSX parameter array */
X
Xopen_vwork()
X{   int i;
X
X    ps_handle=graf_handle(&gl_wchar,&gl_hchar,&gl_wbox,&gl_hbox);
X    for(i=0;i<10;work_in[i++]=1);
X    work_in[10]=2;
X    vs_handle=ps_handle;
X    v_opnvwk(work_in,&vs_handle,work_out);
X    if (vs_handle == 0)
X        panic("v_opnvwk couldn't open window",0L,0L);
X}
X
Xopen_window(kind, name)
Xint kind;
```

```
Xchar *name;
X{   wind_get(0/*whole screen*/,WF_WORKXYWH, &xdesk, &ydesk, &wdesk, &hdesk);
X    wi_handle=wind_create(kind,xdesk,ydesk,wdesk,hdesk);
X
X    wind_set(wi_handle, WF_NAME,name, 0,0);
X    graf_growbox(xdesk+wdesk/2,ydesk+hdesk/
2,gl_wbox,gl_hbox,xdesk,ydesk,wdesk,hdesk);
X    wind_open(wi_handle,xdesk,ydesk,wdesk,hdesk);
X    wind_get(wi_handle,WF_WORKXYWH,&xwork,&ywork,&wwork,&hwork);
X    wwork--;  /* ywork--;/**/ hwork--;
X    set_clip(xwork, ywork, wwork, hwork);
X    wOnewSize(xwork, ywork, wwork, hwork);
X    if (wdesk>500 && 2 != form_alert(1,"[3][16 Colors Required.|\
XSet Preferences to Low|Resolution and Restart.][okay|ignore warning]"))
X         exit();/**/
X}
X
Xopen_rsc(myname, menue)
Xchar *myname;
Xint menue;
X{   if (menue)
X    {    if (rsrc_load(myname)==0) panic("File %s missing!",myname,0L);
X         if (rsrc_gaddr(R_TREE, menue, &p_menu)==0)
X              panic ("AES/rsrc_gaddr() error",0L,0L);
X         if (menu_bar(p_menu, TRUE)==0) panic ("AES/menu_bar() error",0L,0L);
X}   }
X
Xtypedef int CLUT[16][3];
XCLUT (*clut_current);
X
Xw_lClut(new, old)
XCLUT new, old;
X{   int i;
X    clut_current = new;
X    for (i=0; i<16; i++)
X    {    if (old)
X              vq_color(vs_handle, i, 0, old[i]);
X         if (new)
X              vs_color(vs_handle, i, new[i]);
X}   }
X
Xdi_header(dx, str)
Xint dx;
Xchar *str;
X{   set_clip(xdesk, ydesk, wdesk, hdesk);
X/*  v_gtext(vs_handle, ((dx>0)? Wx0: Wx1) +dx, Wy0-3, str);/**/
X    v_gtext(vs_handle, xdesk + (dx<0)*wdesk + dx, ydesk-3, str);
X    set_clip(xwork, ywork, wwork, hwork);
X}
X
Xdi_bg(dx, clr)
Xint dx, clr;
X{   int pxy[4];
```

```
X    set_clip(xdesk, ydesk, wdesk, hdesk);
X    vsf_color(wi_handle, clr);
X    pxy[0] = xdesk+dx; pxy[2] = pxy[0] + 8;
X    pxy[1] = ydesk-10/*3*/; pxy[3] = pxy[1] + 18;
X    v_bar(wi_handle, pxy);
X    set_clip(xwork, ywork, wwork, hwork);
X}
Xset_clip(x,y,w,h)
Xint x,y,w,h;
X{   int clip[4];
X
X    clip[0]=x;    /* set vdi's clip rectangle */
X    clip[1]=y;
X    clip[2]=x+w;
X    clip[3]=y+h;
X    vs_clip(wi_handle,1,clip);
X}
SHAR_EOF
chmod 0600 WNDOWS.C || echo "restore of WNDOWS.C fails"
rm -f s2_seq_.tmp
echo "You have unpacked the last part"
exit 0


From -2017729180275211178
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,79015cd22d6873c5,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-12-20 01:19:49 PST
Path: utzoo!attcan!uunet!lll-winken!lll-lcc!ames!pasteur!ucbvax!agate!saturn!
ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i080: gstlink -- Faster linker for GST object files
Keywords: C
Message-ID: <5804@saturn.ucsc.edu>
Date: 20 Dec 88 09:19:49 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1322
Approved: koreth@ssyx.ucsc.edu

Submitted-by: uunet!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 1, Issue 80
Archive-name: gstlink

[The executable is being posted to the binaries group. -sg]

This program was published in the german magazine ST-Computer (3/88 if i
remember correctly). It is a new (faster, better) linker for the GST BIN format
(used by Lattice-C, Metacomco-Pascal, ...).  It supports the same options like
the default GST-linker (-with, -prog, -sym, -debug, -list, ...) plus several
new ones:
     -sec = generate seperate TEXT, DATA and BSS segments and leave BSS
```

```
            out of the generated program
     -nosec    = generate the same output as the old linker
     -mem = specify internal memory size used for linking the program in
            RAM (yes, therefore it's so fast)
     -buf = specify buffer size used to read in libraries
```

If you use '-sec' the programs will be much smaller than with the original linker from GST.

A friend (of a friend of a ...) hacked this code into the computer so it's poorly documented. But the linker seems to be well debugged, so you should not need to alter the code.

The compiled program uses extended argument passing via either PBP/ARGV (as proposed by Allan Pratt) or ARGV with io_vector like the MWC msh. The use is triggered with the presence or absence of PBP in the environment (and of course the right pointer should be there). I have not included the modified startup.asm for Lattice-C (but if there is some interest, ...) .

- Edgar

```
---------------------- (cut here) ----------------------
/* */
/* */

#include <stdio.h>
#include <osbind.h>
#include <string.h>
#include <ctype.h>

#define FMSIZE     64
#define MAX_LEN    32
#define MAX_PDEF  500
#define MAX_NDEF   20
#define BLEN     1024
#define XMAX       10

#define sgn(x)  ((x)<0?-1:((x)==0?0:1))

typedef enum direct  { data,source,comment,org,section,offset,xdef,xref,
                       define,common,end,eofsy } DIRECT;

typedef char ALFA[MAX_LEN];

typedef struct oper
   {
      short id;
      char  op;
   } OPER;

typedef struct symbol
   {
```

```
      short          length;
      DIRECT         directive;
      char           string[81];
      long           longword;
      short          id;
      char           trunc_rule;
      unsigned char  data_byte;
      short          n_xref;
      OPER           xref_oper[XMAX];
   } SYMBOL;

typedef struct mod_item
   {
      struct mod_item   *mod_next;
      char              mod_name[2];
   } MOD_ITEM;

typedef struct section
   {
      struct section *sec_next;
      short          sec_id;
      char           *sec_start;
      long           sec_length;
      long           sec_oldlen;
      long           sec_fxref;
      long           sec_xptr;
      MOD_ITEM       *sec_module;
      char           sec_name[MAX_LEN];
   } SECTION;

typedef struct xsym
   {
      SECTION        *xsy_sect;
      MOD_ITEM       *xsy_mod;
      long           xsy_value;
      short          xsy_defd;
      struct   xsym  *xsy_left;
      struct   xsym  *xsy_right;
      char           xsy_name[MAX_LEN];
   } XSYMBOL;

typedef struct xoper
   {
      char     xop_oper;
      char     xop_optyp;
      union
      {
         SECTION  *xop_sec;
         XSYMBOL  *xop_sym;
      } xop_ptr;
   } XOPER;

typedef struct xref
```

```
   {
      struct xref    *xref_next;
      long           xref_pos;
      long           xref_abs;
      short          xref_trunc;
      short          xref_ops;
      struct xoper   xref_oper[XMAX];
   } XREF;

extern void       app_xsy(XSYMBOL**,XSYMBOL*);
extern XSYMBOL*   src_xsy(XSYMBOL*,char*);
extern void       app_sec(SECTION**,SECTION*);
extern SECTION*   src_sec(SECTION*,char*);
extern int        read_b();
extern void       printsy();
extern void       nxsy();
extern void       move_up(SECTION*);
extern SECTION*   def_section(SECTION*,char*,short);
extern long       calc_xsy(XSYMBOL*);

SYMBOL   sy;
char     message[]="68000 GST-Format-Linker Version 2.4 / 8.1.1988\n\n";
char     module_name[80];
ALFA     *pdef_name;
ALFA     *ndef_name;
char     input_name[FMSIZE],file_name[FMSIZE],control_name[FMSIZE];
char     listing_name[FMSIZE],program_name[FMSIZE];
short    control_flag,listing_flag,program_flag,debug_flag,symbol_flag;
short    spar_flag;
char     *membot,*memtop,*memstart,*memend,*altstart,*code_ptr,*neustart;
char     *altxref,*debug_start,*debug_end;
long     mem_size,buf_size;
unsigned char *module_buffer,*module_end,*module_ptr,*module_top,*module_max;
SECTION  *curr_sec,*sec_liste,*moved_sec;
SECTION  **sec_lptr=&sec_liste;
MOD_ITEM *mod_liste,*curr_mod;
XSYMBOL  *xsy_liste;
XREF     *xref_liste;
FILE     *list_file;
int      undefd_sym,double_sym,range_err;
short    header[14]={0x601a,0,0,0,0,0,0,0,0,0,0,0,0,0};

char     *errmsg[] = {
   "Out of memory",
   "Program memory too small",
   "Error in binary file",
   "Too many operands in XREF",
   "Illegal section id",
   "Illegal symbol id",
   "ORG encountered",
   "Should not occur",
   "Word or longword at odd address",
   "Cannot write file correctly"
```

```
};

void halt(n)
int n;
{
   printf("Error %2d: %s.\n",n,errmsg[n]);
   printf("Press any key to continue\n");
   gemdos(1);
   exit(1);
}

int strnicmp(x,y,n)
register char           *x,*y;
register unsigned int   n;
{
   if (n<1) return(0);
   while(toupper(*x)==toupper(*y) && *x && *y && --n)
      { x++; y++; }
   return((int)(toupper(*x)-toupper(*y)));
}

void statistic()
{
   SECTION  *s;
   fprintf(list_file,"\n--------------------------\n");
   fprintf(list_file,"SECTION      START   LENGTH\n");
   fprintf(list_file,"--------------------------\n");
   for (s=sec_liste;s!=NULL;s=s->sec_next)
      fprintf(list_file,
         "%-9s %8X %8X\n",s->sec_name,s->sec_start-membot,s->sec_length);
   fprintf(list_file,"--------------------------\n");
}

MOD_ITEM *app_mod(mod_liste,name)
MOD_ITEM **mod_liste;
char     *name;
{
   MOD_ITEM *new;

   new = (MOD_ITEM*)malloc(sizeof(MOD_ITEM)+strlen(name));
   if (new==NULL) halt(0);
   strcpy(new->mod_name,name);
   new->mod_next = *mod_liste;
   *mod_liste = new;
   return(new);
}

void     app_xsy(xsy_liste,xsy_neu)
XSYMBOL  **xsy_liste,*xsy_neu;
{
   int   c;
   if (*xsy_liste==NULL)
   {
```

```
      *xsy_liste=xsy_neu;
      xsy_neu->xsy_left=xsy_neu->xsy_right=NULL;
   }
   else
      if ((c=stricmp(xsy_neu->xsy_name,(*xsy_liste)->xsy_name))<0)
            app_xsy(&(*xsy_liste)->xsy_left,xsy_neu);
      else  app_xsy(&(*xsy_liste)->xsy_right,xsy_neu);
}

XSYMBOL  *src_xsy(xsy_liste,name)
XSYMBOL  *xsy_liste;
char     *name;
{
   int   c;
   if (xsy_liste==NULL) return(NULL);
   if ((c=stricmp(name,xsy_liste->xsy_name))==0) return(xsy_liste);
   if (c<0) return(src_xsy(xsy_liste->xsy_left,name));
   else     return(src_xsy(xsy_liste->xsy_right,name));
   return(NULL);
}

long calc_xsy(s)
XSYMBOL *s;
{
   long  value;

   value=s->xsy_value;
   if (s->xsy_sect!=NULL) value+=s->xsy_sect->sec_start-membot;
   return(value);
}

void debug_table(x)
XSYMBOL  *x;
{
   register char  *p;
   register short i;
   if (x!=NULL)
   {
      if (code_ptr+14>=memtop) halt(1);
      debug_table(x->xsy_left);
      p=x->xsy_name;
      for (i=8;i--;) { *code_ptr++=*p; if (*p) p++; }
      if (x->xsy_sect!=NULL) *((short*)code_ptr)=0xA200;
      else *((short*)code_ptr)=0xA000;
      code_ptr+=2;
      *((long*)code_ptr)=calc_xsy(x);
      code_ptr+=4;
      debug_table(x->xsy_right);
   }
}

void list_xsy(xsy_liste,u_flag)
XSYMBOL  *xsy_liste;
```

```
int      u_flag;
{
   if (xsy_liste!=NULL)
   {
      list_xsy(xsy_liste->xsy_left,u_flag);
      if (u_flag)
      {
         if (!(xsy_liste->xsy_defd&1))
         {
            fprintf(list_file,"Undefined Symbol: '%s'\n",xsy_liste->xsy_name);
            undefd_sym++;
         }
      }
      else
      {
         if (xsy_liste->xsy_defd&1)
         {
            fprintf(list_file,
               "%-20s %08X%c",xsy_liste->xsy_name,calc_xsy(xsy_liste),
                  xsy_liste->xsy_defd&2?' ':'*');
            if (xsy_liste->xsy_sect!=NULL)
               fprintf(list_file," %15s",xsy_liste->xsy_sect->sec_name);
            else fprintf(list_file," %20s"," ");
            if (xsy_liste->xsy_mod!=NULL)
               fprintf(list_file," %15s",xsy_liste->xsy_mod->mod_name);
            fprintf(list_file,"\n");
         }
         else fprintf(list_file,"%-20s undefined\n",xsy_liste->xsy_name);
      }
      list_xsy(xsy_liste->xsy_right,u_flag);
   }
}

void app_sec(sec_liste,sec_neu)
SECTION  **sec_liste,*sec_neu;
{
   if (*sec_liste==NULL)
   {
      *sec_liste=sec_neu;
      sec_neu->sec_next=NULL;
   }
   else  if (sec_neu->sec_id<=(*sec_liste)->sec_id)
            app_sec(&(*sec_liste)->sec_next,sec_neu);
         else
         {
            sec_neu->sec_next=*sec_liste;
            *sec_liste=sec_neu;
         }
}

SECTION* src_sec(sec_liste,name)
SECTION  *sec_liste;
char     *name;
```

```
{
   while(sec_liste!=NULL)
   {
      if (stricmp(sec_liste->sec_name,name)==0) return(sec_liste);
      sec_liste=sec_liste->sec_next;
   }
   return(NULL);
}

unsigned char  inp_buf[BLEN],*buf_ptr,*buf_end;
int            inp_hnd;
int read_b()
{
   if (buf_end-inp_buf<BLEN) return(-1);
   buf_end=inp_buf+Fread(inp_hnd,BLEN,inp_buf);
   buf_ptr=inp_buf;
   return(buf_ptr<buf_end ? (int)*buf_ptr++ : -1);
}

int get_drct()
{
   return(buf_ptr<buf_end ? (int)*buf_ptr++ : read_b());
}

int get_byte()
{
   return(module_ptr<module_end ? (int)*module_ptr++ : get_drct());
}

void nxsy()
{
   int   c;
   if ((c=get_byte())==0xFB)
   {
      register char  *p;
      register int   i;
      switch(get_byte())
      {
         case -1     :  sy.directive=eofsy;
                        sy.length=0;
                        break;
         case 0x01   :  sy.directive=source;
                        sy.length=0;
                        p=sy.string;
                        for(i=get_byte();i--;) *p++=get_byte();
                        *p='\0';
                        break;
         case 0x02   :  sy.directive=comment;
                        sy.length=0;
                        p=sy.string;
                        for(i=get_byte();i--;) *p++=get_byte();
                        *p='\0';
                        break;
```

```
case 0x03   :  sy.directive=org;
               sy.length=0;
               sy.longword=(get_byte()<<24)+(get_byte()<<16)+
                  (get_byte()<<8)+(get_byte());
               break;
case 0x04   :  sy.directive=section;
               sy.length=0;
               sy.id=(get_byte()<<8)+(get_byte());
               break;
case 0x05   :  sy.directive=offset;
               sy.length=0;
               sy.longword=(get_byte()<<24)+(get_byte()<<16)+
                  (get_byte()<<8)+(get_byte());
               break;
case 0x06   :  sy.directive=xdef;
               sy.length=0;
               p=sy.string;
               for(i=get_byte();i--;) *p++=get_byte();
               *p='\0';
               sy.longword=(get_byte()<<24)+(get_byte()<<16)+
                  (get_byte()<<8)+(get_byte());
               sy.id=(get_byte()<<8)+(get_byte());
               break;
case 0x07   :  sy.directive=xref;
               sy.longword=(get_byte()<<24)+(get_byte()<<16)+
                  (get_byte()<<8)+(get_byte());
               sy.trunc_rule=get_byte();
               sy.length=sy.trunc_rule&7;
               for (i=0;(c=get_byte())!=0xFB && i<XMAX;i++)
               {
                  if (c!='+' && c!='-')
                  { printf("Illegal XREF Operator : %c\n",c); halt(2); }
                  sy.xref_oper[i].op=c;
                  sy.xref_oper[i].id=(get_byte()<<8)+(get_byte());
               }
               if (c!=0xFB) halt(3);
               sy.n_xref=i;
               break;
case 0x10   :  sy.directive=define;
               sy.length=0;
               sy.id=(get_byte()<<8)+(get_byte());
               p=sy.string;
               for (i=get_byte();i--;) *p++=get_byte();
               *p='\0';
               break;
case 0x12   :  sy.directive=common;
               sy.length=0;
               sy.id=(get_byte()<<8)+(get_byte());
               break;
case 0x13   :  sy.directive=end;
               sy.length=0;
               break;
case 0xFB   :  sy.directive=data;
```

```
                        sy.length=1;
                        sy.data_byte=0xFB;
                        break;
         default     :  printf("Illegal Directive\n"); halt(2);
                        break;
      }
   }
   else
   {
      if (c==-1)
      {
         sy.directive=eofsy;
         sy.length=0;
      }
      else
      {
         sy.directive=data;
         sy.data_byte=c;
         sy.length=1;
      }
   }
}

void move_up(s)
SECTION *s;
{
   if (memend!=memtop) halt(-1);
   moved_sec=s;
   if (s!=NULL)
      if (s->sec_start!=NULL)
      {
         altstart=memstart;
         memstart=s->sec_start;
         memend=memtop-(altstart-memstart);
         if (altstart>memstart) movmem(memstart,memend,altstart-memstart);
      }
      else moved_sec=NULL;
}

void comment_dir()
{
   fprintf(list_file,"COMMENT: %s\n",sy.string);
   nxsy();
}

void xdef_dir(body_flag)
int   body_flag;
{
   XSYMBOL  *s;

   if ((s=src_xsy(xsy_liste,sy.string))==NULL)
   {
      if ((s=(XSYMBOL*)malloc(sizeof(XSYMBOL)))==NULL) halt(0);
```

```
      strupr(sy.string);
      strncpy(s->xsy_name,sy.string,MAX_LEN-1);
      s->xsy_defd=0;
      s->xsy_mod=NULL;
      app_xsy(&xsy_liste,s);
   }
   if (s->xsy_defd&1)
   {
      fprintf(list_file,"Double defined Symbol: %s\n",sy.string);
      double_sym++;
   }
   else
   {
      if (sy.id)
      {
         if (sy.id>0 || -sy.id>MAX_NDEF) halt(4);
         if ((s->xsy_sect=src_sec(sec_liste,ndef_name[-sy.id]))==NULL)
         {
            if (body_flag) halt(2);
            s->xsy_sect=def_section(sec_liste,ndef_name[-sy.id],sy.id);
         }
      }
      else s->xsy_sect=NULL;
      s->xsy_value=sy.longword;
      if (s->xsy_sect!=NULL) s->xsy_value += s->xsy_sect->sec_oldlen;
      s->xsy_defd |= 1;
      s->xsy_mod = curr_mod;
   }
   nxsy();
}

void define_dir()
{
   strupr(sy.string);
   if (sy.id>0)
   {
      if (sy.id>MAX_PDEF) halt(4);
      strncpy(pdef_name[sy.id],sy.string,MAX_LEN-1);
   }
   else
   {
      if (-sy.id>MAX_NDEF) halt(5);
      strncpy(ndef_name[-sy.id],sy.string,MAX_LEN-1);
   }
   nxsy();
}

SECTION *def_section(sec_liste,name,id)
SECTION  *sec_liste;
char     *name;
short    id;
{
   SECTION  *sec;
```

```
   if (NULL==(sec=(SECTION*)malloc(sizeof(SECTION)))) halt(0);
   strupr(name);
   strncpy(sec->sec_name,name,MAX_LEN-1);
   sec->sec_start  = NULL;
   sec->sec_length = 0;
   sec->sec_oldlen = 0;
   sec->sec_id     = -1;
   sec->sec_module = curr_mod;
   if (!stricmp(sec->sec_name,"DATA")) sec->sec_id=-2;
   if (!stricmp(sec->sec_name,"BSS")) sec->sec_id=-3;
   if (!stricmp(sec->sec_name,"UDATA")) sec->sec_id=-3;
   sec->sec_fxref = NULL;
   sec->sec_xptr  = (long)&sec->sec_fxref;
   app_sec(sec_lptr,sec);
   return(sec);
}

void sec_com_dir()
{
   if (sy.id>=0 || -sy.id>= MAX_NDEF) halt(4);
   if (NULL==(curr_sec=src_sec(sec_liste,ndef_name[-sy.id])))
   {
      curr_sec=def_section(sec_liste,ndef_name[-sy.id],sy.id);
      move_up(curr_sec->sec_next);
      curr_sec->sec_start=memstart;
   }
   else
   {
      move_up(curr_sec->sec_next);
      if (curr_sec->sec_start==NULL) curr_sec->sec_start=memstart;
   }
}

void section_dir()
{
   sec_com_dir();
   code_ptr=neustart=memstart;
   nxsy();
}

void org_dir()
{
   halt(6);
   nxsy();
}

void common_dir()
{
   sec_com_dir();
   neustart=memstart;
   code_ptr=curr_sec->sec_start;
   nxsy();
```

```
}

void data_dir()
{
   if (code_ptr>=memend) halt(1);
   *code_ptr++=sy.data_byte;
   nxsy();
}

void offset_dir()
{
   if (code_ptr>neustart) neustart=code_ptr;
   code_ptr=memstart+sy.longword;
   if (code_ptr>neustart) neustart=code_ptr;
   nxsy();
}

void xref_dir()
{
   XREF     *x;
   XSYMBOL  *xsy;
   SECTION  *sec;
   short    i,xid;

   if ((x=(XREF*)malloc(sizeof(XREF)+(sy.n_xref-XMAX)*sizeof(XOPER)))==NULL)
      halt(0);
   if (curr_sec==NULL) halt(2);
   x->xref_pos    = code_ptr-curr_sec->sec_start;
   x->xref_abs    = sy.longword;
   x->xref_ops    = sy.n_xref;
   x->xref_trunc  = sy.trunc_rule;
   x->xref_next   = NULL;
   for (i=0;i<sy.n_xref;i++)
   {
      x->xref_oper[i].xop_oper=sy.xref_oper[i].op;
      xid=sy.xref_oper[i].id;
      switch(x->xref_oper[i].xop_optyp=sgn(xid))
      {
         case -1  :  if (-xid>MAX_NDEF) halt(4);
                     if ((sec=src_sec(sec_liste,ndef_name[-xid]))==NULL)
                        sec=def_section(sec_liste,ndef_name[-xid],xid);
                     x->xref_oper[i].xop_ptr.xop_sec=sec;
                     if (x->xref_oper[i].xop_oper=='+')
                          x->xref_abs += sec->sec_oldlen;
                     else x->xref_abs -= sec->sec_oldlen;
                     break;
         case  0  :  break;
         case  1  :  if ((xsy=src_xsy(xsy_liste,pdef_name[xid]))==NULL)
                     {
                        if ((xsy=(XSYMBOL*)malloc(sizeof(XSYMBOL)))==NULL)
                           halt(0);
                        strncpy(xsy->xsy_name,pdef_name[xid],MAX_LEN-1);
                        xsy->xsy_defd=0;
```

```
                        xsy->xsy_mod=NULL;
                        app_xsy(&xsy_liste,xsy);
                     }
                     xsy->xsy_defd |= 2;
                     x->xref_oper[i].xop_ptr.xop_sym=xsy;
                     break;
      }
   }
   *((XREF**)curr_sec->sec_xptr)=x;
   curr_sec->sec_xptr=(long)&x->xref_next;
   code_ptr += sy.trunc_rule & 7;
   if (code_ptr>=memend) halt(1);
   nxsy();
}

void header_command()
{
   int   in_header_com=1;

   while (in_header_com)
      switch (sy.directive)
      {
         case comment : comment_dir();break;
         case xdef    : xdef_dir(0);break;
         case define  : define_dir();break;
         default      : in_header_com=0;break;
      }
}

void section_command()
{
   switch (sy.directive)
   {
      case section : section_dir();break;
      case org     : org_dir();break;
      case common  : common_dir();break;
      default      : halt(2);break;
   }
}

void body()
{
   while (sy.directive==data || sy.directive==offset ||
          sy.directive==xdef || sy.directive==xref ||
          sy.directive==define || sy.directive ==comment)
   {
      switch(sy.directive)
      {
         case data     : data_dir();break;
         case offset   : offset_dir();break;
         case xdef     : xdef_dir(1);break;
         case xref     : xref_dir();break;
         case define   : define_dir();break;
```

```
         case comment  : comment_dir();break;
         default       : halt(2);break;
      }
   }
}

void chunk()
{
   SECTION *s;

   while (sy.directive==xdef || sy.directive==comment ||
          sy.directive==define) header_command();
   if (sy.directive==section || sy.directive==org ||
       sy.directive==common)
   {
      section_command();
      body();
      if (((long)code_ptr&1) && code_ptr>=neustart)
      {
         if (code_ptr>=memend) halt(1);
         *code_ptr++='\0';
      }
      if (code_ptr>neustart) neustart=code_ptr;
      curr_sec->sec_length+=neustart-memstart;
      if (altstart!=NULL)
      {
         if (altstart>memstart) movmem(memend,neustart,altstart-memstart);
         for(s=moved_sec;s!=NULL;s=s->sec_next)
            if (s->sec_start!=NULL) s->sec_start += neustart-memstart;
         memend=memtop;
         neustart+=altstart-memstart;
         altstart=NULL;
      }
      memstart=neustart;
   }
}

void module()
{
   SECTION  *sec;
   short    i;

   if (sy.directive!=source) halt(2);
   curr_mod=app_mod(&mod_liste,sy.string);
   strcpy(module_name,sy.string);
   nxsy();
   while (sy.directive==xdef || sy.directive==comment ||
          sy.directive==define || sy.directive==section ||
          sy.directive==org || sy.directive==common) chunk();
   if (sy.directive!=end) halt(2);
   if (listing_flag) fprintf(list_file,"%-12.12s:",module_name);
   i=0;
   for (sec=sec_liste;sec!=NULL;sec=sec->sec_next)
```

```
   {
      if (listing_flag)
      {
         if (i++>3) { fprintf(list_file,"\n"); i=0; }
         fprintf(list_file," %8.8s=%08X",
                 sec->sec_name,sec->sec_length-sec->sec_oldlen);
      }
      sec->sec_oldlen=sec->sec_length;
   }
   if (listing_flag) fprintf(list_file,"\n");
   strcpy(module_name,"NO MODULE");
   nxsy();
}

void calc_xref(x,c,modname)
XREF  *x;
char  *c,*modname;
{
   short i;
   long  value;

   value = x->xref_abs;
   c += x->xref_pos;
   /* printf("XREF at %8X %8X",c-membot,value); */
   for (i=0;i<x->xref_ops;i++)
   {
      /* printf("%c",x->xref_oper[i].xop_oper); */
      switch (x->xref_oper[i].xop_optyp)
      {
         case -1 : if (x->xref_oper[i].xop_oper=='+')
                        value+=x->xref_oper[i].xop_ptr.xop_sec->sec_start
                                -membot;
                   else value-=x->xref_oper[i].xop_ptr.xop_sec->sec_start
                               -membot;
                   /* printf("%s/%x",x->xref_oper[i].xop_ptr.xop_sec->sec_name,
                               x->xref_oper[i].xop_ptr.xop_sec->sec_start); */
                   break;
         case  0 : if (x->xref_oper[i].xop_oper=='+')
                        value+=c-membot;
                   else value-=c-membot;
                   printf("&");
                   break;
         case  1 : if (x->xref_oper[i].xop_oper=='+')
                         value+=calc_xsy(x->xref_oper[i].xop_ptr.xop_sym);
                   else  value-=calc_xsy(x->xref_oper[i].xop_ptr.xop_sym);
                   /* printf("%s"/%x",x->xref_oper[i].xop_ptr.xop_sym->xsy_name,
                               calc_xsy(x->xref_oper[i].xop_ptr.xop_sym)); */
                   break;
      }
      /* printf("\n"); */
   }
   if (c<membot || c>memstart) halt(2);
   if (x->xref_trunc & 32) value -=c-membot;
```

```
switch(x->xref_trunc & 7)
{
   case 4 : if (((long)c)&1) halt(1); *((long*)c)=value; break;
   case 2 : if (((long)c)&1) halt(1); *((short*)c)=value;
            if (x->xref_trunc & 8)
            {
               if (value<-32768L || value>32767L)
               {
                  range_err++;
                  printf("XREF.W-value out of range in module '%s'\n",
                          modname);
               }
            }
            else
            {
               if (value>65535L)
               {
                  range_err++;
                  printf("XREF.UW-value out of range in module '%s'\n",
                          modname);
               }
            }
            break;
   case 1 : *c=value;
            if (x->xref_trunc & 8)
            {
               if (value<-128L || value>128L)
               {
                  range_err++;
                  printf("XREF.B-value out of range in module '%s'\n",
                          modname);
               }
            }
            else
            {
               if (value>255L)
               {
                  range_err++;
                  printf("XREF.UB-value out of range in module '%s'\n",
                          modname);
               }
            }
            break;
   default: halt(2);
}
if (x->xref_trunc & 64)
{
   if (altxref==NULL)
   {
      if (code_ptr>=memtop-4) halt(1);
      altxref=c;
      *((long*)code_ptr)=c-membot;
      code_ptr+=4;
```

```
      }
      else
      {
         while (c-altxref>254)
         {
            altxref+=254;
            if (code_ptr>=memtop) halt(1);
            *code_ptr++='\001';
         }
         if (code_ptr>=memtop) halt(1);
         *code_ptr++=c-altxref;
         altxref=c;
      }
   }
}

void all_xrefs(sec_liste)
SECTION  *sec_liste;
{
   XREF  *x;

   while (sec_liste!=NULL)
   {
      x=(XREF*)sec_liste->sec_fxref;
      while (x!=NULL)
      {
         calc_xref(x,sec_liste->sec_start,sec_liste->sec_module->mod_name);
         x=x->xref_next;
      }
      sec_liste=sec_liste->sec_next;
   }
   if (altxref==NULL)
   {
      if (code_ptr>=memtop-8) halt(1);
      *((long*)code_ptr)=0; code_ptr+=4;
      *((long*)code_ptr)=0; code_ptr+=4;
   }
   else
   {
      if (code_ptr>=memtop) halt(1);
      *code_ptr++='\0';
   }
}

void init_mem()
{
   pdef_name=(ALFA*)malloc(MAX_PDEF*sizeof(ALFA));
   ndef_name=(ALFA*)malloc(MAX_NDEF*sizeof(ALFA));
   membot=(char*)malloc(mem_size);
   module_buffer=(char*)malloc(buf_size);
   if (membot==NULL || module_buffer==NULL ||
       ndef_name==NULL || pdef_name==NULL) halt(0);
```

```
   memtop=membot+mem_size;
   memstart=membot;
   memend=memtop;
   altstart=NULL;
   module_top=module_buffer+buf_size;
   module_max=module_ptr=module_end=module_buffer;
}


void make_ext(make_name,name,ext)
char  *make_name,*name,*ext;
{
   char  oldtext[FMSIZE];

   stcgfe(oldtext,name);
   if (!*oldtext) strmfe(make_name,name,ext);
   else           strcpy(make_name,name);
}

void command_line(argc,argv)
int   argc;
char  *argv[];
{
   int   i=2;
   int   x;

   if (argc<2) { printf("No Filename specified\n"); exit(1); }
   strcpy(file_name,argv[1]);
   strmfe(input_name,file_name,"BIN");
   strmfe(listing_name,file_name,"MAP");
   strmfe(control_name,file_name,"LNK");
   strmfe(program_name,file_name,"PRG");
   listing_flag=0;
   control_flag=0;
   program_flag=1;
   debug_flag=0;
   symbol_flag=0;
   spar_flag=0;
   i=2;
   if (argc>i) if (*argv[i]!='-')
      { make_ext(control_name,argv[i++],"LNK"); control_flag=1; }
   if (argc>i) if (*argv[i]!='-')
      { make_ext(listing_name,argv[i++],"MAP"); listing_flag=1; }
   if (argc>i) if (*argv[i]!='-') make_ext(program_name,argv[i++],"PRG");
   for (;i<argc;i++)
   {
      if (!stricmp(argv[i],"-NOLIST")) {listing_flag=0; continue;}
      if (!stricmp(argv[i],"-NODEBUG")) {debug_flag=0; continue;}
      if (!stricmp(argv[i],"-NOPROG")) {program_flag=0; continue;}
      if (!stricmp(argv[i],"-DEBUG")) {debug_flag=1; continue;}
      if (!stricmp(argv[i],"-SYM")) {symbol_flag=1; continue;}
      if (!stricmp(argv[i],"-NOSYM")) {symbol_flag=0; continue;}
      if (!stricmp(argv[i],"-SEC")) {spar_flag=1; continue;}
```

```
      if (!stricmp(argv[i],"-NOSEC")) {spar_flag=0; continue;}
      if (!stricmp(argv[i],"-WITH"))
      {
         if (i+1<argc) if (*argv[i+1]!='-')
            make_ext(control_name,argv[++i],"LNK");
         control_flag=1;
         continue;
      }
      if (!stricmp(argv[i],"-LIST"))
      {
         if (i+1<argc) if (*argv[i+1]!='-')
            make_ext(listing_name,argv[++i],"LST");
         listing_flag=1;
         continue;
      }
      if (!stricmp(argv[i],"-PROG"))
      {
         if (i+1<argc) if (*argv[i+1]!='-')
            make_ext(program_name,argv[++i],"PRG");
         program_flag=1;
         continue;
      }
      if (!stricmp(argv[i],"-MEM"))
      {
         if (i+1<argc) if (*argv[i+1]!='-')
         {
            x=atoi(argv[++i]);
            if (x>2 && x<800) mem_size=x*1024;
         }
         continue;
      }
      if (!stricmp(argv[i],"-BUF"))
      {
         if (i+1<argc) if (*argv[i+1]!='-')
         {
            x=atoi(argv[++i]);
            if (x>2 && x<800) buf_size=x*1024;
         }
         continue;
      }
      printf("Invalid Option: '%s'\n",argv[i]);
   }
}

int test_module()
{
   register int   c;
   int            end_test=0;
   int            result=0;
   register short i;
   char           string[80];
   register char  *p;
   XSYMBOL        *s;
```

```
module_end=module_ptr=module_buffer;

do
{
   c=get_drct(); if (c<0) halt(2);
   if (module_end+128>module_top) halt(9);
   *module_end++=c;
   if (c==0xFB)
   {
      c=get_drct(); *module_end++=c;
      switch(c)
      {
         case   -1 : halt(2); break;
         case 0xFB : break;
         case 0x01 :
         case 0x02 : i=get_drct(); *module_end++=i;
                     while(i--) *module_end++=get_drct();
                     break;
         case 0x03 :
         case 0x05 : *module_end++=get_drct();
                     *module_end++=get_drct();
                     *module_end++=get_drct();
                     *module_end++=get_drct();
                     break;
         case 0x12 :
         case 0x04 : *module_end++=get_drct();
                     *module_end++=get_drct();
                     break;
         case 0x06 : p=string;
                     i=get_drct(); *module_end++=i;
                     while (i--)
                        { *p=get_drct(); *module_end++=*p++; }
                     *p='\0';
                     *module_end++=get_drct();
                     *module_end++=get_drct();
                     *module_end++=get_drct();
                     *module_end++=get_drct();
                     *module_end++=get_drct();
                     *module_end++=get_drct();
                     s=src_xsy(xsy_liste,string);
                     if (s!=NULL) if (!(s->xsy_defd&1))
                        end_test=result=1;
                     break;
         case 0x07 : *module_end++=get_drct();
                     *module_end++=get_drct();
                     *module_end++=get_drct();
                     *module_end++=get_drct();
                     *module_end++=get_drct();
                     while (1)
                     {
                        c=get_drct(); *module_end++=c;
                        if (c==0xFB || c==-1) break;
```

```
                        *module_end++=get_drct();
                        *module_end++=get_drct();
                     }
                     break;
            case 0x10 : *module_end++=get_drct();
                        *module_end++=get_drct();
                        i=get_drct(); *module_end++=i;
                        while (i--) *module_end++=get_drct();
                        break;
            case 0x13 : end_test=1;
                        break;
            default   : halt(2);
         }
      }
   } while (!end_test);
   if (module_end>module_max) module_max=module_end;
   return(result);
}

void link_file(name,lib_mode)
char  *name;
int   lib_mode;
{
   inp_hnd=Fopen(name,0);
   buf_ptr=buf_end=inp_buf+BLEN;
   strcpy(module_name,"*NO MODULE");
   if (inp_hnd<0)
   {
      printf("Cannot open binary file: '%s'\n",name);
      exit(1);
   }

   nxsy();
   while (sy.directive!=eofsy)
   {
      if (lib_mode)
      {
         if(!test_module())
         {
            module_end=module_ptr=module_buffer;
            nxsy();
         }
         else module();
      }
      else module();
   }
   Fclose(inp_hnd);
}

void write_prog()
{
   int      handle;
   int      n,h;
```

```
SECTION  *sec;
char     *start,*endcode;

if ((handle=Fcreate(program_name,0))<0)
{
   printf("Cannot open %s for writing\n"); halt(10);
}
*((long*)&header[7])=debug_end-memstart;
n=Fwrite(handle,28,header); if (n!=28) halt(10);

if (spar_flag)
{
   h=1;
   sec=sec_liste;
   start=membot;

   while (sec->sec_id>-2 && sec->sec_next!=NULL) sec=sec->sec_next;
   if (sec->sec_id==-2)
   {
      endcode=sec->sec_start;
      *((long*)&header[h])=endcode-start;
      if (endcode-start)
      {
         n=Fwrite(handle,endcode-start,start);
         if (n!=endcode-start) halt(10);
      }
      h=3;
      start=endcode;
   }

   while (sec->sec_id>-3 && sec->sec_next!=NULL) sec=sec->sec_next;
   if (sec->sec_id==-3)
   {
      endcode=sec->sec_start;
      {
         *((long*)&header[h])=endcode-start;
         if (endcode-start)
         {
            n=Fwrite(handle,endcode-start,start);
            if (n!=endcode-start) halt(10);
         }
      }
      h=5;
      start=endcode;
   }

   /* Rest in (h)-Section schreiben */
   endcode=debug_start;
   *((long*)&header[h])=endcode-start;
   if (h<5)
   {
      if (endcode-start)
      {
```

```
            n=Fwrite(handle,endcode-start,start);
            if (n!=endcode-start) halt(10);
         }
      }
   }
   else
   {
      start=membot;
      endcode=debug_start;
      *((long*)&header[1])=debug_start-membot;
      if (endcode-start)
      {
         n=Fwrite(handle,endcode-start,start);
         if (n!=endcode-start) halt(10);
      }
   }
   /* Symboltabelle und Relocation Table schreiben */
   *((long*)&header[7])=debug_end-debug_start;
   start=debug_start;
   endcode=code_ptr;
   if (endcode-start)
   {
      n=Fwrite(handle,endcode-start,start);
      if (n!=endcode-start) halt(10);
   }
   Fseek(0L,handle,0);
   n=Fwrite(handle,28,header); if (n!=28) halt(10);
   if (Fclose(handle)<0) halt(10);
}

void main(argc,argv)
int   argc;
char  *argv[];
{

   char  line[80],*name;
   FILE  *fp;
   int   lib_mode;
   int   err_code=0;

   printf(message);
   mem_size=100*1024; buf_size=32*1024;
   double_sym=undefd_sym=range_err=0;
   command_line(argc,argv);
   mod_liste=NULL;
   list_file=stdout;
   if (listing_flag)
   {
      list_file=fopen(listing_name,"w");
      if (list_file==NULL)
      {
         printf("Cannot open list_file '%s' for writing\n"); exit(1);
      }
```

```
   if (list_file!=stdout) fprintf(list_file,message);
}

init_mem();
sec_liste=curr_sec=moved_sec=NULL;
xsy_liste=NULL;
xref_liste=NULL;
altxref=NULL;

if (control_flag)
{
   fp=fopen(control_name,"r");
   if (fp==NULL)
   {
      printf("Cannot open control_file : '%s'\n",control_name);
      exit(1);
   }
   while (!feof(fp))
   {
      if ((name=fgets(line,80,fp))==NULL) *line='\0';
      for (name=line; *name!='\n' && *name; name++);
      *name='\0';
      if (!*line) continue;
      if (*line=='*') continue;
      name=NULL;
      lib_mode=0;
      if (!strnicmp(line,"INPUT",5)) name=line+5;
      else if (!strnicmp(line,"LIBRARY",7)) { name=line+7; lib_mode=1; }
      if (name==NULL) printf("Invalid control line : %s\n",line);
      else
      {
         name=stpblk(name);
         if (*name=='*') strmfe(name,file_name,"BIN");
         else            make_ext(name,name,"BIN");
         link_file(name,lib_mode);
      }
   }
   fclose(fp);
}
else link_file(input_name,0);
code_ptr=memstart;
debug_start=code_ptr;
if (debug_flag) debug_table(xsy_liste);
debug_end=code_ptr;
all_xrefs(sec_liste);

statistic();
fprintf(list_file,"Program length   = %8X\n",memstart-membot);
fprintf(list_file,"Symbol table     = %8X\n",debug_end-memstart);
fprintf(list_file,"Relocation table = %8X\n",code_ptr-debug_end);
fprintf(list_file,"--------------------\n");
fprintf(list_file,
   "Memory Usage     = %7d%%\n",(code_ptr-membot)*100/mem_size);
```

```
   fprintf(list_file,
      "Buffer Usage     = %7d%%\n",(module_max-module_buffer)*100/buf_size);
   fprintf(list_file,"--------------------\n");
   list_xsy(xsy_liste,1);
   if (program_flag) write_prog();
   if (symbol_flag)
   {
      fprintf(list_file,"\nSymbol Table:\n");
      fprintf(list_file,"-------------\n");
      list_xsy(xsy_liste,0);
      fprintf(list_file,"\n");
   }
   if (undefd_sym)
   {
      if (list_file!=stdout)
         fprintf(list_file,"Undefined Symbols: %8d\n",undefd_sym);
      printf("Undefined Symbols: %8d\n",undefd_sym);
      err_code=1;
   }
   if (double_sym)
   {
      if (list_file!=stdout)
         fprintf(list_file,"Multiply defined : %8d\n",double_sym);
      printf("Multiply defined : %8d\n",double_sym);
      err_code=1;
   }
   if (range_err)
   {
      if (list_file!=stdout)
         fprintf(list_file,"Range errors     : %8D\n",range_err);
      printf("Range errors     : %8D\n",range_err);
      err_code=1;
   }

   printf("\nLink completed\n");
   if (list_file!=stdout) fprintf(list_file,"\nLink completed\n");
   if (listing_flag) fclose(list_file);
   exit(err_code);
}
```


```
From -8054988141806379533
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,ad11211e31c04b15,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-12-27 00:07:48 PST
Path: utzoo!attcan!uunet!husc6!cmcl2!rutgers!ucsd!ucbvax!agate!saturn!
ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i081: mx2v230 -- Multitasking kernel with utilities  part01/08
Keywords: shar, Modula-2
Message-ID: <5865@saturn.ucsc.edu>
```

Date: 27 Dec 88 08:07:48 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 896
Approved: koreth@ssyx.ucsc.edu

Submitted-by: madsen@sask.usask.ca (Jorgen Madsen)
Posting-number: Volume 1, Issue 81
Archive-name: mx2v230/part01

[The binaries have been posted to comp.binaries.atari.st. -sg]

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
# This is part 1 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
#
#    Run the following text with /bin/sh to create:
#      ACC.MOD
#      ATOMIC.DEF
#      ATOMIC.MOD
#      BITSTUFF.DEF
#      BITSTUFF.MOD
#      CLI.MOD
#      COM.MOD
#      KILLLP.MOD
#      LOGIN.DEF
#      LOGIN.MOD
#      LP.MOD
#      MX2.MOD
#      MX2LIB.C
#      MX2NET.MOD
#      MX2NOTES.TXT
#      NETAUX.MOD
#      NETMIDI.MOD
#      NETWORK.DEF
#      NETWORK.MOD
#      NEWSYS.DEF
#      NEWSYS.MOD
#      PASSWD.MOD
#      PIPE.DEF
#      PIPE.MOD
#      PS.MOD
#      REDIR.MOD
#      RUN.MOD
#      SCANNER.DEF
#      SCANNER.MOD
#      SPAWN.MOD
#      SPOOLER.MOD
#      SQ.MOD
#      SUBMIT.MOD
#      SUBMITA.MOD
#      SUBMITM.MOD
```

```
#      SYS.MOD
#      SYSCALL.DEF
#      SYSCALL.MOD
#      XA.MOD
#      XMODEM.DEF
#      XMODEM.MOD
#      XMODEMR.MOD
#      XMODEMT.MOD
#      XS.MOD
#
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
sed 's/^X//' << 'SHAR_EOF' > ACC.MOD &&
X
X(*              Copyright 1987,1988 fred brooks LogicTek        *)
X(*                                                              *)
X(*                                                              *)
X(*   First Release                      12/8/87-FGB             *)
X(* Added code to tell the context switcher the value of the     *)
X(* ACC Super Stack to help it with multitasking gem programs    *)
X(*                                      3/9/88-FGB              *)
X(*                                                              *)
X
XMODULE acc;
X(*$T-,$S-*)
XFROM ATOMIC          IMPORT VERSION;
XFROM SYSTEM          IMPORT ADDRESS,ADR;
XFROM GEMDOS          IMPORT Super,ConOut;
XFROM AESApplications IMPORT ApplInitialise;
XFROM AESMenus        IMPORT MenuRegister;
XFROM AESEvents       IMPORT EventMultiple;
XFROM GEMAESbase      IMPORT AccessoryOpen;
XFROM AESForms        IMPORT FormAlert;
X
XVAR
X  applID                          : INTEGER;  (* desk application ID *)
X  menuID,waittime                 : INTEGER;  (* menu ID *)
X  Msg                             : ARRAY [0..16] OF INTEGER;
X  handle, events, x,emask,b       : INTEGER;
X  ssv,stacksave                   : ADDRESS;
X  superstack [144H]               : ADDRESS;
X  mx2run                          : BOOLEAN;
X
X
XBEGIN
X  (* initialise application & install desk accessory *)
X  applID := ApplInitialise();
X  menuID := MenuRegister(applID,VERSION);
X  emask:=48; (* timer and message *)
X  waittime:=10000; (* check every 10 seconds to see if mx2 is running *)
X  mx2run:=FALSE;
```

```
X
X  ssv:=0;
X  Super(ssv);           (* in super mode *)
X  superstack:=ssv;      (* save superstack value for context switcher *)
X  stacksave:=ssv;
X  Super(ssv);           (* return to gem stack space *)
X  LOOP
X    events := EventMultiple(emask,0,0,0,
X                            0,0,0,0,0,
X                            0,0,0,0,0,
X                            ADR(Msg),
X                            waittime,0,
X                            x,x,x,x,
X                            x,x);
X  IF NOT mx2run THEN
X     ssv:=0;
X     Super(ssv);           (* in super mode *)
X     IF superstack#stacksave THEN
X        waittime:=0; (* set timer to zero *)
X        mx2run:=TRUE;
X     END;
X     Super(ssv);           (* return to gem stack space *)
X  END;
X  IF Msg[0]=AccessoryOpen THEN
X        IF emask=48 THEN b:=1 ELSE b:=2 END;
X        ConOut(7c);
X        x:=FormAlert(b,"[2][Fred Brooks LogicTek| |     Switch|MX2 Desk
Accessory][ON|OFF]");
X        Msg[0]:=0;
X        IF x=1 THEN
X           emask:=48; (* message and timer *)
X        ELSE
X           emask:=16; (* just message *)
X        END;
X  END;
X  END;
XEND acc.
SHAR_EOF
chmod 0600 ACC.MOD || echo "restore of ACC.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > ATOMIC.DEF &&
XDEFINITION MODULE ATOMIC;
X
X(*              Copyright 1987,1988 fred brooks LogicTek        *)
X(*                                                              *)
X(*                                                              *)
X(*   First Release                      12/8/87-FGB             *)
X(* Added variable parm to StartProcess to pass info to process  *)
X(* in currentprocess.gemsave[15]        1/1/88-FGB              *)
X(*                                                              *)
X(* Remove monitor priority. Each routine the switches processes *)
X(* must be protected from all interrupts by the IntEnd and      *)
X(* IntBegin calls. If this is not done correctly the system     *)
X(* system will bomb.                    4/4/88-FGB              *)
```

```
X(*                                                                  *)
X
XFROM    SYSTEM  IMPORT  ADDRESS,BYTE;
XFROM    NEWSYS  IMPORT  PROCESS;
XFROM    Strings IMPORT  String;
X
XCONST   MAGIC            =       22261; (* somebodys b-day *)
X        buflength        =       1024;
X        OTOS             =       LONGCARD(602CH); (* Old rom TOS *)
X        MTOS             =       LONGCARD(87CEH); (* New rom MEGA TOS *)
X        OLDDATE          =       11201985H;
X        NEWDATE          =       04221987H;
X        VERSION          =       "  MX2 V2.3.0"; (* mx2 version number *)
XTYPE
X        Gvectype         =       POINTER TO ADDRESS;
X        buffer  =       ARRAY [0..buflength] OF BYTE;
X        PIPE    =       RECORD
X                pipename:       String;
X                buf     :       buffer;
X                bufsize :       CARDINAL;
X                bufhead :       CARDINAL;
X                buftail :       CARDINAL;
X                cnt     :       CARDINAL;
X        END;
X        pipeptr =       POINTER TO PIPE;
X(*      these devices are defined for use with the MX2 port command
X        dev0 thru dev7 are user defineable device drivers. By loading
X        the addresses of the correct procedures into the DeviceTable for
X        in, out and status custom devices may be defined.
X        DeviceTable[dev0].bconstat:=Mybconstat;
X        DeviceTable[dev0].bcostat:=Mybcostat;
X        DeviceTable[dev0].bconin:=Mybconin;
X        DeviceTable[dev0].bconout:=Mybconout;
X        Will setup dev0 for use with the port command.
X
X        sys port 4 -0 -0
X        will cause the process with the PID of 4 to use dev0
X*)
X        devicetype =    (printer,aux,con,midi,null,dev0,dev1,dev2,dev3,
X                         dev4,dev5,dev6,dev7);
X        devstattype     =       PROCEDURE(): BOOLEAN;
X        devintype       =       PROCEDURE(): LONGCARD;
X        devouttype      =       PROCEDURE(CHAR);
X
X        devblocktype    =       RECORD
X                bconstat        :       devstattype;
X                bcostat         :       devstattype;
X                bconin          :       devintype;
X                bconout         :       devouttype;
X        END;
X        devtabletype    =       ARRAY [ORD(dev0)..ORD(dev7)] OF devblocktype;
X
XTYPE            SIGNAL          =       POINTER TO ProcessDescriptor;
```

```
X               GEMTYPE         =       ARRAY [0..0FFH] OF ADDRESS;
X               pipetype        =       ARRAY [0..31] OF pipeptr;
X       (* pipe names starting with mx2 or MX2 are reserved by the
X          system *)
X(* This is the control record for all system process. Variable
X   "currentprocess" points to the currently running process. The ready
X   queue is implemented as a linked list round robin. *)
X
X(* The gemsave array contains important system vectors that are switched
X   with every process. This allows programs to intercept trap vectors
X   such as the GEMDOS TRAP 1 and still to allow the other programs
X   to run correctly.       current vectors saved are
X                           gemsave[0]      gem workspace pointer
X                           gemsave[1]      LINEA   VECTOR    28H
X                           gemsave[2]      GEMDOS  TRAP 1    84H
X                           gemsave[3]      GSX,GEM TRAP 2    88H
X                           gemsave[4]      BIOS    TRAP 13   B4H
X                           gemsave[5]      XBIOS   TRAP 14   B8H
X                           gemsave[6]      linef   VECTOR    2CH
X                           gemsave[7]      level2  VECTOR    68H
X                           gemsave[8]      level4  VECTOR    70H
X                           gemsave[9]      _shell_p         4F6H
X                           gemsave[11]     wait value (* V2.2.0 *)
X                           gemsave[12]     wait mask  (* V2.2.0 *)
X                           gemsave[13]     sleep timer variable
X                           gemsave[14]     timer flag variable
X                           gemsave[15]     init parameter for process
X
X                           misc[0]         priority work variable
X*)
X
X       ProcessDescriptor =
X              RECORD
X                       name    :       String;   (* process name *)
X                       pid     :       INTEGER;  (* process id   *)
X                       cor     :       PROCESS;  (* process coroutine *)
X                       pc      :       ADDRESS;  (* process PC *)
X                       sr      :       CARDINAL; (* process SR *)
X                       ssp     :       ADDRESS;  (* process super SR *)
X                       biosval :       ADDRESS;  (* bios call save ptr *)
X                       retval  :       ADDRESS;
X                       biosave :       ARRAY [0..225] OF CARDINAL;
X                       gemsave :       ARRAY [0..15]  OF ADDRESS;
X                       termvec :       ADDRESS;
X                       wsp     :       ADDRESS;  (* process work space *)
X                       wspsize :       LONGCARD; (* size of work space *)
X                       next    :       SIGNAL;   (* ptr to next process *)
X                       ppid    :       INTEGER;  (* parent id *)
X                       tmpcor  :       PROCESS;  (* cor save variable *)
X                       ready   :       BOOLEAN;  (* process run flag *)
X                       active  :       BOOLEAN;  (* process alive flag *)
X                       intflag :       BOOLEAN;  (* process switched *)
X                       date    :       CARDINAL; (* date started *)
```

```
X                       time    :       CARDINAL; (* time started *)
X                       tick    :       LONGCARD; (* total 200hz clocks *)
X                       slice   :       LONGCARD; (* restart time 200hz *)
X                       pri     :       INTEGER;  (* set length of run *)
X                       flags   :       ARRAY [0..2] OF LONGCARD; (* var *)
X(* flags[0] is default drive of process that called the kernel
X   flags[1] is address of process to be added to ready list of kernel
X   flags[2] is size of process stack or workspace
X*)
X                       ipname  :       String;   (* program name *)
X                       iptail  :       String;   (* program command *)
X                     ipenvstr  :       String;   (* program env string *)
X                       return  :       INTEGER;  (* function return *)
X                       errno   :       INTEGER;  (* error number *)
X                       bpsave  :       GEMTYPE;  (* save for gemdos *)
X                       Iport   :       devicetype;
X                       Oport   :       devicetype;
X                       waitloc :       POINTER TO LONGCARD; (* V2.2.0 *)
X                       misc    :       ARRAY [0..8] OF INTEGER;
X               END;
X
X(* Vector 144H points to variable that contains this record *)
X(* look at SYS.MOD for an example of use of these variables *)
X
X     sysrequesttype =
X               RECORD
X                  req          : BOOLEAN;
X                  pid          : INTEGER;
X                  magic        : LONGCARD;
X               END;
X
X     spinttype =
X               RECORD
X                  proc         : PROC;
X                  pid          : INTEGER;
X                  data         : ADDRESS;
X               END;
X
X     StartProcesstype           = PROCEDURE(VAR PROC,VAR LONGCARD,
X                                            VAR INTEGER,VAR String,
X                                            VAR ADDRESS);
X     SwapProcesstype            = PROCEDURE;
X     TermProcesstype            = PROCEDURE(VAR INTEGER);
X     NextPidtype                = PROCEDURE(): INTEGER;
X     SleepProcesstype           = PROCEDURE(VAR INTEGER);
X     WakeupProcesstype          = PROCEDURE(VAR INTEGER);
X     ChangeProcessPrioritytype  = PROCEDURE(VAR INTEGER,VAR INTEGER);
X     MultiBegintype             = PROCEDURE;
X     MultiEndtype               = PROCEDURE;
X     DozeProcesstype            = PROCEDURE(VAR INTEGER,VAR LONGCARD);
X     WaitProcesstype            = PROCEDURE(VAR INTEGER,VAR ADDRESS,
X                                            VAR LONGCARD, VAR LONGCARD,
X                                            VAR LONGCARD);
```

```
X
X       sysvariable =
X              RECORD
X                 currentprocess  : POINTER TO SIGNAL;
X                 MULTI           : POINTER TO BOOLEAN;
X                 slicebegin      : POINTER TO LONGCARD;
X                 contextswitch   : POINTER TO LONGCARD;
X                 command         : POINTER TO String;
X                 request         : POINTER TO sysrequesttype;
X                 CRON            : POINTER TO PROC;
X                 spintenable     : POINTER TO BITSET;
X                 spintmask       : POINTER TO BITSET;
X                 spint           : POINTER TO ARRAY [0..15] OF spinttype;
X                 bpsave          : POINTER TO GEMTYPE;
X                 pipes           : POINTER TO pipetype;
X                 sysmemsize      : POINTER TO LONGCARD;
X                 gemsaveGvec     : POINTER TO Gvectype;
X                 StartProcess    : StartProcesstype;
X                 SwapProcess     : SwapProcesstype;
X                 TermProcess     : TermProcesstype;
X                 NextPid         : NextPidtype;
X                 SleepProcess    : SleepProcesstype;
X                 WakeupProcess   : WakeupProcesstype;
X                 ChangeProcessPriority : ChangeProcessPrioritytype;
X                 MultiBegin      : MultiBegintype;
X                 MultiEnd        : MultiEndtype;
X                 DozeProcess     : DozeProcesstype;
X                 WaitProcess     : WaitProcesstype;
X                 CronActive      : POINTER TO BOOLEAN;
X                 DeviceTable     : POINTER TO devtabletype;
X              END;
X
XVAR             currentprocess              : SIGNAL;
X                MULTI                       : BOOLEAN;
X                slicebegin                  : LONGCARD;
X                contextswitch               : LONGCARD;
X                command                     : String;
X                request                     : sysrequesttype;
X                CRON                        : PROC;
X                spintenable                 : BITSET;
X                spintmask                   : BITSET;
X                spint                       : ARRAY [0..15] OF spinttype;
X                bpsave                      : GEMTYPE;
X                pipes                       : pipetype;
X                sysmemsize                  : LONGCARD;
X                ROMDATE [0fc0018H]          : LONGCARD;
X                gemsaveGvec                 : Gvectype;
X                CronActive                  : BOOLEAN;
X                DeviceTable                 : devtabletype;
X                (* A spint is a software interrupt. It is called by
X                   setting it's bit in the spint enable mask. Spints
X                   0-3 are reserved by the system   *)
X                (* The CRON procedure variable is here to put your
```

```
X                   own procedure into the init scheduler in MX2 this
X                   procedure will be run every 30 seconds. It is setup to
X                   the dummy procedure at first. to setup your own
X                   get sysvar then (sysvar.CRON:='your procedure') *)
X
X(* start the master context switch interrupt and setup system variables *)
XPROCEDURE Initsked;
X
X(* setup kernal to begin the system startup setup all buffers and
X   procedures for MX2 *)
XPROCEDURE InitProcesses;
X
X(* shutdown kernal and restore to normal gem state *)
X(* this currently does not work *)
XPROCEDURE EndProcesses;
X
X(* create a process for the MX2 system, place in the scheduler ready
X   list, start new process *)
XPROCEDURE StartProcess(VAR P: PROC;
X                       VAR n: LONGCARD;
X                       VAR priority: INTEGER;
X                       VAR pn: String;
X                       VAR parm: ADDRESS);
X                       (* parm is the init parameter to be placed in
X                          gemsave[15]  *)
X
X(* store the currentprocess and switch to next process in ready list *)
XPROCEDURE SwapProcess;
X
X(* end process and remove it from the ready list, free memory used by
X   process *)
XPROCEDURE TermProcess(VAR id: INTEGER);
X
X(* return the next pid the system will use for a new process *)
XPROCEDURE NextPid(): INTEGER;
X
X(* tell scheduler not to run this process but keep in memory *)
XPROCEDURE SleepProcess(VAR id: INTEGER);
X
X(* tell scheduler to sleep for msec 1000th of a second  *)
XPROCEDURE DozeProcess(VAR id: INTEGER; VAR msec: LONGCARD);
X
X(* Sleep process until the contents of 'loc' equals 'value AND mask'.
Xmsec is the timeout value in milliseconds if set the 0 it waits forever *)
XPROCEDURE WaitProcess(VAR id: INTEGER; VAR loc: ADDRESS;
X                      VAR value,mask,msec: LONGCARD);
X
X(* tell scheduler to start running this process again if it was sleeping
X   before *)
XPROCEDURE WakeupProcess(VAR id: INTEGER);
X
X(* V0.7 change process priority *)
XPROCEDURE ChangeProcessPriority(VAR id: INTEGER; VAR pri: INTEGER);
```

```
X
X(* turn on the scheduler interrupt, and start normal process switching *)
XPROCEDURE MultiBegin;
X
X(* turn off the scheduler interrupt, used to stop process switching in
X   section of code that should not be swapped out *)
XPROCEDURE MultiEnd;
X
X(* check if flag is TRUE and lock out other process from changing it's
X   state until procedure is over *)
XPROCEDURE CheckFlag(VAR flag: BOOLEAN): BOOLEAN;
X
X(* set flag to TRUE and lock out other process from changing it's
X   state until procedure is over *)
XPROCEDURE SetFlag(VAR flag: BOOLEAN);
X
X(* set flag to FALSE and lock out other process from changing it's
X   state until procedure is over *)
XPROCEDURE ResetFlag(VAR flag: BOOLEAN);
X
X(* check if flag is TRUE then set flag to FALSE while locking out other
X   process from changing it's state until procedure is over *)
XPROCEDURE CheckResetFlag(VAR flag: BOOLEAN): BOOLEAN;
X
XEND             ATOMIC.
X
SHAR_EOF
chmod 0600 ATOMIC.DEF || echo "restore of ATOMIC.DEF fails"
sed 's/^X//' << 'SHAR_EOF' > ATOMIC.MOD &&
X
X(*              Copyright 1987,1988 fred brooks LogicTek        *)
X(*                                                              *)
X(*                                                              *)
X(*   First Release                      12/8/87-FGB             *)
X(*                                                              *)
X(* Modified TermProcess to also kill all the children processes *)
X(* of the parent. Changed FindProcess not to find zombies       *)
X(*                                      12/11/87-FGB            *)
X(*                                                              *)
X(* Added variable parm to StartProcess to pass info to process  *)
X(* in currentprocess.gemsave[15]        1/1/88-FGB              *)
X(* The PID of the new process will be returned in variable parm *)
X(*                                      2/24/88-FGB             *)
X(*                                                              *)
X(* Added DozeProcess to allow timed a sleep of processes        *)
X(*                                      2/21/88-FGB             *)
X(*                                                              *)
X(* Remove monitor priority. Each routine the switches processes *)
X(* must be protected from all interrupts by the IntEnd and      *)
X(* IntBegin calls. If this is not done correctly the system     *)
X(* system will bomb.                    4/4/88-FGB              *)
X(*                                                              *)
X
```

```
X
X(*$T-,$S-,$A+ *)
XIMPLEMENTATION MODULE          ATOMIC;
X
XFROM  SYSTEM  IMPORT   ADR,TSIZE,
X                       CODE,SETREG,ADDRESS,REGISTER;
XFROM  NEWSYS  IMPORT   NEWPROCESS,PROCESS,TRANSFER,IOTRANSFER;
XFROM  XBIOS   IMPORT   SuperExec;
XFROM  BitStuff IMPORT   LAnd;
X
XFROM  GEMDOS  IMPORT   GetTime,GetDate,Super;
X
XFROM  Storage IMPORT   ALLOCATE,DEALLOCATE;
X
XFROM  Strings IMPORT   String;
X
XTYPE    workspace       =       ARRAY [0..254] OF CARDINAL;
X        savetype        =       POINTER TO ARRAY [0..22] OF CARDINAL;
X        sigs            =       (sleep,wakeup,terminate,trace,routine,
X                                 program,wait,gem,tos);
X        CPFlagtype      =       SET OF sigs;
X        CPFlagptrtype   =       POINTER TO CPFlagtype;
X
XCONST   trapvec         =       110H; (* vector on IOTRANSFER *)
X        offsetvec       =       140H; (* offset IOTRANSFER vector *)
X        intnum          =       4;    (* interrupt number on MFP *)
X
XVAR        s0,s1,s2,schedproc,s,
X           initproc,lastproc    :       SIGNAL;
X           kin                  :       ARRAY [0..32] OF INTEGER;
X           wsp0,wsp1,wsp2,wsp3  :       workspace;
X           wsp,bios             :       ADDRESS;
X           etimer [400H]        :       ADDRESS;
X           biospointer [4a2H]   :       ADDRESS;
X           ptermvec     [408H]  :       ADDRESS;
X           trap [trapvec]       :       ADDRESS;
X           memtop [436H]        :       ADDRESS;
X           flock [43eH]         :       CARDINAL;
X           hz200  [4baH]        :       LONGCARD;
X           temphz200,TICKS      :       LONGCARD;
X           sysvector [144H]     :       POINTER TO sysvariable;
X           accsuperstack        :       ADDRESS;
X           sysvar               :       sysvariable;
X           linea [28H]          :       ADDRESS;
X           gemdos [84H]         :       ADDRESS;
X           gsxgem [88H]         :       ADDRESS;
X           tbios  [0b4H]        :       ADDRESS;
X           xbios  [0b8H]        :       ADDRESS;
X           linef  [2cH]         :       ADDRESS;
X           level2 [68H]         :       ADDRESS;
X           level4 [70H]         :       ADDRESS;
X           shellp [4F6H]        :       ADDRESS;
X           oldtrap,oldtermvec   :       ADDRESS;
```

```
X           OldEtimer,sspval     :       ADDRESS;
X           p,cotick,x,t,temp    :       PROCESS;
X           processesid,I        :       INTEGER;
X           zombie               :       BOOLEAN;
X           children,highpri     :       INTEGER;
X           savefrom,saveto      :       savetype;
X           CPFlagptr            :       CPFlagptrtype;
X
X        (* --------------  BEGIN -------------- *)
X
XPROCEDURE       StartProcess(VAR P        : PROC;
X                             VAR n        : LONGCARD;
X                             VAR priority : INTEGER;
X                             VAR pn       : String;
X                             VAR parm     : ADDRESS);
XBEGIN
X        IntEnd;
X        SuperExec(UpdatecurrentProc);
X        ALLOCATE(wsp,n);
X        IF wsp=NIL THEN
X           currentprocess^.errno:=12;
X           IntBegin;
X           RETURN;
X        END;
X        DEC(sysmemsize,n);
X        s1:=currentprocess;
X        s0:=currentprocess;
X
X        IF processesid>0 THEN    (* search only after setup of SCHED *)
X        s0:=schedproc;               (* set to sched process *)
X         LOOP                    (* find zombie process *)
X          s0:=s0^.next;
X          IF s0^.pid=1 THEN      (* zombie not found in list *)
X             EXIT;
X          END;
X          FindZombieProcess(zombie);
X          IF zombie THEN EXIT END;
X         END;
X        END; (* if processesid>1 *)
X
X        IF zombie THEN
X           currentprocess:=s0;
X        ELSE
X           ALLOCATE(currentprocess,TSIZE(ProcessDescriptor));
X           IF currentprocess=NIL THEN
X              s1^.errno:=12;
X              IntBegin;
X              RETURN;
X           END;
X           DEC(sysmemsize,TSIZE(ProcessDescriptor));
X           INC(processesid);
X           currentprocess^.pid:=processesid;
X           currentprocess^.next:=initproc; lastproc^.next:=currentprocess;
```

```
X           lastproc:=currentprocess;
X        END;
X
X        WITH currentprocess^ DO
X             name:=pn;
X             ppid:=request.pid;
X             ready:=TRUE; active:=TRUE;
X             tick:=0;
X             IF priority<1 THEN priority:=1 END;
X             IF priority>10 THEN priority:=10 END;
X             pri:=priority;
X             misc[0]:=pri;
X             IF pri>highpri THEN highpri:=pri END;
X             GetTime(time);
X             GetDate(date);
X             gemsave[13]:=0; (* set timer to zero *)
X             gemsave[14]:=0; (* set all flags to zero *)
X             gemsave[15]:=parm;
X             parm:=ADDRESS(currentprocess^.pid);
X             Iport:=s1^.Iport;
X             Oport:=s1^.Oport;
X        END;
X        IF currentprocess^.pid>1 THEN
X           FindProcess(request.pid,s2);
X           s2^.return:=currentprocess^.pid;
X        ELSE
X           currentprocess^.return:=0;
X        END;
X        SuperExec(SetSlice);
X        currentprocess^.slice:=temphz200;
X        currentprocess^.wsp:=wsp;
X        currentprocess^.wspsize:=n;
X        NEWPROCESS(P,wsp,n,currentprocess^.cor);
X        SuperExec(intbiosptr);
X        s1^.errno:=0;
X        INC(contextswitch);            (* update switch counter *)
X        TRANSFER(s1^.cor,currentprocess^.cor);
X        IntBegin;
XEND     StartProcess;
X
XPROCEDURE       TermProcess(VAR id: INTEGER);
XBEGIN
X        IF id<2 THEN RETURN END;
X        IntEnd;
X
X        FindProcess(id,s2);
X        IF s2=NIL THEN
X           currentprocess^.errno:=3;
X           IntBegin;
X           RETURN;
X        END;
X        currentprocess^.errno:=0;
X
```

```
X        s2^.active:=FALSE;      (* set flag *)
X        IF s2^.wsp#NIL THEN
X           DEALLOCATE(s2^.wsp,s2^.wspsize);
X           INC(sysmemsize,s2^.wspsize);
X        END;
X        s2^.wsp:=NIL;           (* set TO NIL to make zombie process *)
X        s0:=currentprocess; (* set to the parent process *)
X
X        kin[0]:=id;
X        REPEAT (* loop thru process list and find all related processes *)
X        FindChildProcess(kin[0],s1);
X        WHILE s1#NIL DO
X           INC(children);
X           kin[children]:=s1^.pid;
X           s1^.active:=FALSE;      (* set flag *)
X           s1^.gemsave[14]:=0; (* reset all flags *)
X           IF s1^.wsp#NIL THEN
X              DEALLOCATE(s1^.wsp,s1^.wspsize);
X              INC(sysmemsize,s1^.wspsize);
X           END;
X           s1^.wsp:=NIL;           (* set TO NIL to make zombie process *)
X           currentprocess:=s1;     (* set currentprocess to terminated *)
X           SuperExec(IncSlice);    (* update cpu time for terminated *)
X           FindChildProcess(kin[0],s1);
X        END;
X        kin[0]:=kin[children];
X        DEC(children);
X        UNTIL children<0;
X        currentprocess:=s0;
X
X        currentprocess^.errno:=0;
X        s1:=currentprocess;     (* save currentprocess *)
X        currentprocess:=s2;     (* set currentprocess to terminated *)
X        SuperExec(UpdatecurrentProc);  (* update cpu time for terminated *)
X
X
X        currentprocess:=schedproc;  (* set to transfer to the sched *)
X
X        IF s1^.pid#id THEN
X           temp:=s1^.cor; (* currentprocess different than terminated *)
X        ELSE
X           temp:=t;     (* process the same so set up dummy process *)
X        END;
X
X        IF s1^.pid#1 THEN
X           SuperExec(UpdateProc);
X           TRANSFER(temp,currentprocess^.cor); (* do context switch *)
X        END;
X        IntBegin;
XEND             TermProcess;
X
XPROCEDURE       NextPid(): INTEGER;
XBEGIN
```

```
X        s1:=currentprocess;
X        s0:=currentprocess;
X
X        IF processesid>0 THEN    (* search only after setup of SCHED *)
X        s0:=schedproc;              (* set to sched process *)
X         LOOP                   (* find zombie process *)
X          s0:=s0^.next;
X          IF s0^.pid=1 THEN      (* zombie not found in list *)
X             EXIT;
X          END;
X          FindZombieProcess(zombie);
X          IF zombie THEN EXIT END;
X         END;
X        END; (* if processesid>1 *)
X
X        IF zombie THEN
X           RETURN s0^.pid;
X        ELSE
X           RETURN processesid+1;
X        END;
XEND             NextPid;
X
XPROCEDURE       SleepProcess(VAR id: INTEGER);
XBEGIN
X        IF id<2 THEN RETURN END;
X
X        IntEnd;
X        FindProcess(id,s2);
X        IF s2=NIL THEN
X           currentprocess^.errno:=3;
X           IntBegin;
X           RETURN;
X        END;
X        currentprocess^.errno:=0;
X
X        IF (s2^.wsp#NIL) AND (NOT s2^.active) THEN IntBegin; RETURN END;
X        IF s2^.wsp#NIL THEN
X           s2^.active:=FALSE;      (* set flag *)
X           CPFlagptr:=ADR(s2^.gemsave[14]);
X           INCL(CPFlagptr^,sleep);
X           s2^.gemsave[13]:=0;
X        END;
X
X        s1:=currentprocess;     (* save currentprocess *)
X        SuperExec(UpdatecurrentProc);    (* update cpu time for sleeper *)
X
X        currentprocess:=schedproc;  (* set to transfer to the sched *)
X
X        IF s1^.pid#1 THEN
X           s1^.ready:=FALSE; currentprocess^.ready:=TRUE;
X           SuperExec(UpdateProc);
X           TRANSFER(s1^.cor,currentprocess^.cor); (* do context switch *)
X        END;
```

```
X        IntBegin;
XEND             SleepProcess;
X
XPROCEDURE       DozeProcess(VAR id: INTEGER; VAR msec : LONGCARD);
XBEGIN
X        IF id<2 THEN RETURN END;
X
X        IntEnd;
X        FindProcess(id,s2);
X        IF s2=NIL THEN
X           currentprocess^.errno:=3;
X           IntBegin;
X           RETURN;
X        END;
X        currentprocess^.errno:=0;
X
X        IF (s2^.wsp#NIL) AND (NOT s2^.active) THEN
X           IntBegin;
X           RETURN;
X        END;
X        IF s2^.wsp#NIL THEN
X           s2^.active:=FALSE;      (* set flag *)
X           CPFlagptr:=ADR(s2^.gemsave[14]);
X           INCL(CPFlagptr^,sleep);
X           IF msec#0 THEN
X                SuperExec(SetSlice);
X                s2^.gemsave[13]:=ADDRESS(temphz200+(msec DIV 5));
X           ELSE
X                s2^.gemsave[13]:=0;
X           END;
X        END;
X
X        s1:=currentprocess;     (* save currentprocess *)
X        SuperExec(getbiosptr);
X        SuperExec(UpdatecurrentProc);    (* update cpu time for sleeper *)
X
X        currentprocess:=schedproc;  (* set to transfer to the sched *)
X
X        IF s1^.pid#1 THEN
X           s1^.ready:=FALSE; currentprocess^.ready:=TRUE;
X           SuperExec(UpdateProc);
X           TRANSFER(s1^.cor,currentprocess^.cor); (* do context switch *)
X        END;
X        IntBegin;
XEND             DozeProcess;
X
XPROCEDURE       WaitProcess(VAR id: INTEGER; VAR loc: ADDRESS;
X                            VAR value,mask,msec : LONGCARD);
XBEGIN
X        IF id<2 THEN RETURN END;
X
X        IntEnd;
X        FindProcess(id,s2);
```

```
X        IF s2=NIL THEN
X           currentprocess^.errno:=3;
X           IntBegin;
X           RETURN;
X        END;
X        currentprocess^.errno:=0;
X
X        IF (s2^.wsp#NIL) AND (NOT s2^.active) THEN
X           IntBegin;
X           RETURN;
X        END;
X        IF s2^.wsp#NIL THEN
X           s2^.active:=FALSE;      (* set flag *)
X           CPFlagptr:=ADR(s2^.gemsave[14]);
X           INCL(CPFlagptr^,sleep);
X           INCL(CPFlagptr^,wait);
X           s2^.waitloc:=loc;
X           s2^.gemsave[11]:=ADDRESS(value);
X           s2^.gemsave[12]:=ADDRESS(mask);
X           IF msec#0 THEN
X                SuperExec(SetSlice);
X                s2^.gemsave[13]:=ADDRESS(temphz200+(msec DIV 5));
X           ELSE
X                s2^.gemsave[13]:=0;
X           END;
X        END;
X
X        s1:=currentprocess;     (* save currentprocess *)
X        SuperExec(UpdatecurrentProc);    (* update cpu time for sleeper *)
X
X        currentprocess:=schedproc;  (* set to transfer to the sched *)
X
X        IF s1^.pid#1 THEN
X           s1^.ready:=FALSE; currentprocess^.ready:=TRUE;
X           SuperExec(UpdateProc);
X           TRANSFER(s1^.cor,currentprocess^.cor); (* do context switch *)
X        END;
X        IntBegin;
XEND             WaitProcess;
X
XPROCEDURE       WakeupProcess(VAR id: INTEGER);
XBEGIN
X        IF id<2 THEN RETURN END;
X
X        IntEnd;
X        FindProcess(id,s2);
X        IF s2=NIL THEN
X           currentprocess^.errno:=3;
X           IntBegin;
X           RETURN;
X        END;
X
X        IF (s2^.wsp#NIL) AND s2^.active THEN
```

```
X           currentprocess^.errno:=0;
X           IntBegin;
X           RETURN; (* already awake *)
X        END;
X
X        IF s2^.wsp#NIL THEN
X           s2^.active:=TRUE;      (* set flag *)
X           CPFlagptr:=ADR(s2^.gemsave[14]);
X           EXCL(CPFlagptr^,sleep);
X           currentprocess^.errno:=0;
X        ELSE
X           currentprocess^.errno:=3;
X        END;
X        IntBegin;
XEND             WakeupProcess;
X
X(* V0.7 *)
XPROCEDURE       ChangeProcessPriority(VAR id: INTEGER; VAR pri: INTEGER);
XBEGIN
X        IF id<2 THEN RETURN END;
X
X        IntEnd;
X        FindProcess(id,s2);
X        IF s2=NIL THEN
X           currentprocess^.errno:=3;
X           IntBegin;
X           RETURN;
X        END;
X
X        IF pri<1  THEN pri:=1 END;
X        IF pri>10 THEN pri:=10 END;
X        s2^.pri:=pri;
X        IntBegin;
XEND             ChangeProcessPriority;
X
XPROCEDURE       InitProcesses;
XBEGIN
X        CRON:=dummy;
X        ALLOCATE(currentprocess,TSIZE(ProcessDescriptor));
X        DEC(sysmemsize,TSIZE(ProcessDescriptor));
X        processesid:=0;
X        initproc:=currentprocess;
X        lastproc:=initproc;
X        WITH currentprocess^ DO
X             next:=currentprocess;
X             ready:=FALSE;
X             active:=FALSE;
X             name:="INIT";
X             GetTime(time);
X             GetDate(date);
SHAR_EOF
echo "End of part 1, continue with part 2"
echo "2" > s2_seq_.tmp
```

```
exit 0


From 3123960150458411313
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,be2b4356f3ec2fdf,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-12-27 00:07:56 PST
Path: utzoo!attcan!uunet!husc6!cmcl2!rutgers!ucsd!ucbvax!agate!saturn!
ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i082: mx2v230 -- Multitasking kernel with utilities  part02/08
Keywords: shar, Modula-2
Message-ID: <5866@saturn.ucsc.edu>
Date: 27 Dec 88 08:07:56 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1066
Approved: koreth@ssyx.ucsc.edu

Submitted-by: madsen@sask.usask.ca (Jorgen Madsen)
Posting-number: Volume 1, Issue 82
Archive-name: mx2v230/part02

#!/bin/sh
# this is part 2 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file ATOMIC.MOD continued
#
CurArch=2
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> ATOMIC.MOD
X        END;
X        NEWPROCESS(dummy,ADR(wsp1),TSIZE(workspace),x);
X        NEWPROCESS(dummy,ADR(wsp2),TSIZE(workspace),p);
X        NEWPROCESS(dummy,ADR(wsp3),TSIZE(workspace),t);
X        currentprocess^.wsp:=ADR(wsp2);
X        currentprocess^.cor:=p;
XEND             InitProcesses;
X
XPROCEDURE       EndProcesses;
XBEGIN
X        s0:=currentprocess;
X        REPEAT
X                currentprocess:=s0^.next;
```

```
X               s0:=currentprocess;
X       UNTIL currentprocess^.pid=0;
X       SuperExec(restorebiosptr);
X       INC(contextswitch);            (* update switch counter *)
X       TRANSFER(currentprocess^.cor,currentprocess^.cor);
XEND    EndProcesses;
X
X(* This is the scheduler which is called by newtrap *)
XPROCEDURE      sched;
XBEGIN
X       IOTRANSFER(cotick,x,ADDRESS(trapvec)); (* V1.1 baud rate timer *)
X       LOOP                                   (* interrupt not used by ST *)
X            currentprocess^.cor:=x;
X            currentprocess^.ready:=FALSE;
X            currentprocess^.intflag:=TRUE; (* process interrupted *)
X            INC(contextswitch);            (* update switch counter *)
X            currentprocess:=s0; (* s0 set to new in newtrap procedure *)
X            currentprocess^.ready:=TRUE;   (* process to start *)
X            x:=currentprocess^.cor;
X            IOTRANSFER(cotick,x,ADDRESS(offsetvec)); (* offset vector so not *)
X       END;                                        (* to overwrite the     *)
XEND    sched;                                      (* newtrap vector       *)
X
XPROCEDURE      SwapProcess; (* swap out processes *)
XBEGIN
X       IntEnd;
X       SuperExec(UpdatecurrentProc);
X       s0:=currentprocess;
X       currentprocess^.ready:=FALSE;
X       LOOP (* find next process *)
X         currentprocess:=currentprocess^.next;
X         IF currentprocess^.active AND (currentprocess^.wsp#NIL) THEN
X            EXIT
X         END;
X       END; (* loop *)
X       currentprocess^.ready:=TRUE;
X       IF s0^.pid#currentprocess^.pid THEN
X          SuperExec(UpdateProc);
X          TRANSFER(s0^.cor,currentprocess^.cor);
X       END;
X       IntBegin;
XEND    SwapProcess;
X
X(*$P- *)
XPROCEDURE      UpdateProc; (* update all currentprocess values *)
XBEGIN
X       currentprocess^.slice:=hz200;
X       ClrInt;
X       SetTimer;
X       setbiosptr;
X       INC(contextswitch);            (* update switch counter *)
X       CODE(4e75H); (* rts *)
XEND    UpdateProc;
```

```
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE       UpdatecurrentProc;
XBEGIN
X        getbiosptr;
X        IncSlice;
X        CODE(4e75H); (* rts *)
XEND             UpdatecurrentProc;
X(*$P+ *)
X
XPROCEDURE       dummy; (* dummy procedure to make newprocess *)
XBEGIN
XEND             dummy;
X
XPROCEDURE       d1(): BOOLEAN;
XBEGIN
XEND             d1;
X
XPROCEDURE       d2(c: CHAR);
XBEGIN
XEND             d2;
X
XPROCEDURE       d3(): LONGCARD;
XBEGIN
XEND             d3;
X(* check to see if process s0 is a ZOMBIE, if so return true else false *)
XPROCEDURE       FindZombieProcess(VAR zombie: BOOLEAN);
XBEGIN
X          IF (NOT s0^.active) AND (s0^.pid#0) AND (s0^.wsp=NIL) THEN
X             zombie:=TRUE; (* found one *)
X          ELSE
X             zombie:=FALSE; (* nada it's alive! *)
X          END;
XEND             FindZombieProcess;
X
XPROCEDURE       FindProcess(VAR pid: INTEGER; VAR fp: SIGNAL);
XBEGIN
X        s:=schedproc;
X        LOOP                    (* find  process id *)
X          s:=s^.next;
X          IF (s^.pid=pid) AND (s^.wsp#NIL) THEN     (* found id *)
X             fp:=s;
X             EXIT;
X          END;
X          IF s^.pid=schedproc^.pid THEN      (* id not found in list *)
X             fp:=NIL;
X             EXIT;
X          END;
X        END;
XEND     FindProcess;
X
XPROCEDURE       FindChildProcess(VAR pid: INTEGER; VAR fp: SIGNAL);
```

```
XBEGIN
X        s:=schedproc;
X        LOOP                    (* find  process child id *)
X          s:=s^.next;
X          IF (s^.ppid=pid) AND (s^.wsp#NIL) THEN     (* found id *)
X             fp:=s;
X             EXIT;
X          END;
X          IF s^.pid=schedproc^.pid THEN      (* id not found in list *)
X             fp:=NIL;
X             EXIT;
X          END;
X        END;
XEND     FindChildProcess;
X
X(*$P- *)
XPROCEDURE   IncSlice; (* UPDATE cpu time  *)
XBEGIN
X        INC(currentprocess^.tick,hz200-currentprocess^.slice);
X        currentprocess^.slice:=hz200;
X        CODE(4e75H); (* rts *)
XEND         IncSlice;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE   SetSlice; (* start timeslice  *)
XBEGIN
X        temphz200:=hz200;
X        CODE(4e75H); (* rts *)
XEND         SetSlice;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE   SetTimer; (* setup number of ticks before switch V1.1  *)
XBEGIN
X        TICKS:=hz200+LONGCARD(currentprocess^.pri*10);
X        CODE(4e75H);            (* rts *)
XEND         SetTimer;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE   ClrInt; (* V1.2 change to int on mfp *)
XBEGIN
X        CODE(8b9H,intnum,0ffffH,0fa11H); (* BCLR intnum, ISRB *)
X        CODE(8b9H,intnum,0ffffH,0fa15H); (* BCLR intnum, MASKB *)
X        CODE(4e75H);               (* rts *)
XEND         ClrInt;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE   EnableInterrupt;
XBEGIN                           (* setup sched vector interrupt *)
X        SETREG(8,0fffffa15H);   (* load a0 with adr of mfp reg *)
```

```
X        CODE(8d0H,intnum);  (* set int mask bit *)
X        CODE(4e75H);            (* rts *)
XEND         EnableInterrupt;
X(*$P+ *)
X
XPROCEDURE       MultiBegin;
XBEGIN
X        SuperExec(ClrInt);
X        SuperExec(SetTimer);
X        MULTI:=TRUE;
XEND             MultiBegin;
X
XPROCEDURE       MultiEnd;
XBEGIN
X        MULTI:=FALSE;
X        SuperExec(ClrInt);
XEND             MultiEnd;
X
XPROCEDURE       IntBegin;
XVAR     ssv     :       ADDRESS;
XBEGIN
X        ssv:=0H;
X        Super(ssv);
X        CODE(46fcH,2300H);  (* start interrupts *)
X        Super(ssv);
XEND             IntBegin;
X
XPROCEDURE       IntEnd;
XVAR     ssv     :       ADDRESS;
XBEGIN
X        ssv:=0H;
X        Super(ssv);
X        CODE(46fcH,2700H);  (* stop interrupts *)
X        Super(ssv);
XEND             IntEnd;
X
X(*$P- *)
XPROCEDURE       getbiosptr;
XBEGIN
X        WITH currentprocess^ DO
X                biosval:=biospointer+46;
X                termvec:=ptermvec;
X                gemsave[0]:=gemsaveGvec^;
X                gemsave[1]:=linea;
X                gemsave[2]:=gemdos;
X                gemsave[3]:=gsxgem;
X                gemsave[4]:=tbios;
X                gemsave[5]:=xbios;
X                gemsave[6]:=linef;
X                gemsave[7]:=level2;
X                gemsave[8]:=level4;
X                gemsave[9]:=shellp;
X        END;
```

```
X        CODE(4e75H);               (* rts *)
XEND             getbiosptr;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE       setbiosptr;
XBEGIN
X        WITH currentprocess^ DO
X        savefrom:=ADDRESS(biospointer);
X        saveto:=ADDRESS(biosval-46);
X        saveto^:=savefrom^;
X        biospointer:=biosval-46;
X        ptermvec:=termvec;
X        gemsaveGvec^:=gemsave[0];
X        linea:=gemsave[1];
X        gemdos:=gemsave[2];
X        gsxgem:=gemsave[3];
X        tbios:=gemsave[4];
X        xbios:=gemsave[5];
X        linef:=gemsave[6];
X        level2:=gemsave[7];
X        level4:=gemsave[8];
X        shellp:=gemsave[9];
X        END;
X        CODE(4e75H);               (* rts *)
XEND             setbiosptr;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE       restorebiosptr;
XBEGIN
X        savefrom:=biospointer;
X        saveto:=bios;
X        saveto^:=savefrom^;
X        biospointer:=bios;
X        ptermvec:=oldtermvec;
X        CODE(4e75H);               (* rts *)
XEND             restorebiosptr;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE       intbiosptr;
XBEGIN
X        WITH currentprocess^ DO
X        savefrom:=ADDRESS(biospointer);
X        saveto:=ADR(biosave[199]);
X        saveto^:=savefrom^;
X        biosval:=ADR(biosave[199]);
X        biospointer:=biosval;
X        termvec:=ptermvec;
X        gemsave[0]:=gemsaveGvec^;
X        gemsave[1]:=linea;
X        gemsave[2]:=gemdos;
```

```
X        gemsave[3]:=gsxgem;
X        gemsave[4]:=tbios;
X        gemsave[5]:=xbios;
X        gemsave[6]:=linef;
X        gemsave[7]:=level2;
X        gemsave[8]:=level4;
X        gemsave[9]:=shellp;
X        END;
X        CODE(4e75H);              (* rts *)
XEND             intbiosptr;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE       newtrap; (* IOTRANSFER executes this code before its *)
XBEGIN                    (* normal vector. *)
X
X        CODE(46fcH,2700H);  (* stop interrupts *)
X        CODE(8b9H,intnum,0ffffH,0fa11H); (* BCLR intnum, ISRB *)
X        CODE(8b9H,intnum,0ffffH,0fa15H); (* BCLR intnum, MASKB *)
X        CODE(817H,5);                 (* BTST 5, (a7) check supermode *)
X        CODE(6700H+2);                (* BEQ.S over rte *)
X        CODE(4e73H);                  (* RTE supermode return *)
X        CODE(48e7H,0fffeH);   (* save regs movem  *)
X        CODE(204fH);    (* move.l ssp,a0 *)
X        currentprocess^.ssp:=REGISTER(8)+66;
X
X        IF (currentprocess^.ssp#sspval)
X           AND (currentprocess^.ssp#accsuperstack) THEN
X           CODE(4cdfH,7fffH); (* restore regs movem *)
X           CODE(4e73H);       (* rte *)
X        ELSE
X
X           IncSlice;   (* cpu time update *)
X           s0:=currentprocess;
X           LOOP                    (* find next process store in s0 *)
X              s0:=s0^.next;
X              WITH s0^ DO
X                IF (gemsave[14]#0) THEN (* If flags set then process *)
X                   CPFlagptr:=ADR(gemsave[14]);
X
X                   IF sleep IN CPFlagptr^ THEN (* sleep flag *)
X                      IF (gemsave[13]#0) AND
X                         (ADDRESS(hz200) >= gemsave[13]) THEN
X                         active:=TRUE;
X                         gemsave[13]:=0;
X                         EXCL(CPFlagptr^,wait);  (* clear wait flag *)
X                         EXCL(CPFlagptr^,sleep); (* clear sleep flag *)
X                      END;
X                   END;
X                   IF wait IN CPFlagptr^ THEN (* wait flag *)
X                      IF (waitloc^ = LONGCARD(LAnd(gemsave[11],gemsave[12])))
THEN
X                         active:=TRUE;
```

```
X                        gemsave[13]:=0;
X                        EXCL(CPFlagptr^,wait);  (* clear wait flag *)
X                        EXCL(CPFlagptr^,sleep); (* clear sleep flag *)
X                     END;
X                  END;
X
X               END;
X               IF (active) AND (wsp#NIL) THEN
X                  IF misc[0]>=highpri THEN
X                     IF pri>highpri THEN highpri:=pri END;
X                     misc[0]:=pri;
X                     EXIT;
X                  ELSE
X                     INC(misc[0]); (* age process til it's ready to run *)
X                  END;
X               END;
X            END; (* with *)
X         END; (* end LOOP *)
X
X      (* Swap GEM pointers for the processes *)
X         WITH currentprocess^ DO
X         gemsave[0]:=gemsaveGvec^;
X         gemsave[1]:=linea;
X         gemsave[2]:=gemdos;
X         gemsave[3]:=gsxgem;
X         gemsave[4]:=tbios;
X         gemsave[5]:=xbios;
X         gemsave[6]:=linef;
X         gemsave[7]:=level2;
X         gemsave[8]:=level4;
X         gemsave[9]:=shellp;
X         biosval:=biospointer;
X         termvec:=ptermvec;
X         END;
X         WITH s0^ DO
X         biospointer:=biosval;
X         ptermvec:=termvec;
X         gemsaveGvec^:=gemsave[0];
X         linea:=gemsave[1];
X         gemdos:=gemsave[2];
X         gsxgem:=gemsave[3];
X         tbios:=gemsave[4];
X         xbios:=gemsave[5];
X         linef:=gemsave[6];
X         level2:=gemsave[7];
X         level4:=gemsave[8];
X         shellp:=gemsave[9];
X         END;
X
X         SetTimer;
X         s0^.slice:=hz200; (* set next cycle start *)
X         SETREG(8,oldtrap);  (* move IOTRANSFER trap adr *)
X         CODE(43faH,10); (* lea 12(pc),a1 *)
```

```
X           CODE(2288H); (* move.l a0,(a1) *)
X           CODE(4cdfH,7fffH); (* restore regs movem *)
X           CODE(4ef9H,0,0) (* jmp back to routine *)
X        END;
XEND     newtrap;
X(*$P+ *)
X
X(*$P-,$S- *)
XPROCEDURE       NewEtimer;
XBEGIN
X        CODE(48e7H,0fffeH);   (* save regs movem  *)
X           IF MULTI THEN
X              CODE(207cH,0ffffH,0fa15H);  (* load a0 with adr of mfp reg *)
X              CODE(8d0H,intnum);      (* set int mask bit *)
X              CODE(5188H);         (* subq.l 8 adj a0 to adr of mfp reg *)
X              CODE(5988H);         (* subq.l 4 adj a0 to adr of mfp reg *)
X              CODE(8d0H,intnum);      (* set int enable bit *)
X           END;
X
X           SETREG(8,OldEtimer);  (* move trap adr *)
X           CODE(43faH,6); (* lea 8(pc),a1 *)
X           CODE(2288H); (* move.l a0,(a1) *)
X           CODE(4ef9H,0,0); (* jmp back to routine *)
XEND     NewEtimer;
X(*$P+,$S- *)
X
X(*$P- *) (*save trap and set up flag trap for IOTRANSFER to run my code *)
XPROCEDURE       settrap;     (* before executing the IOTRANSFER.        *)
XBEGIN
X        CODE(46fcH,2700H);  (* stop interrupts  V1.1 use 200hz clock *)
X        TICKS:=10;
X        OldEtimer:=etimer+4;
X        etimer:=ADDRESS(NewEtimer);
X        EnableInterrupt;
X        MULTI:=FALSE;
X        oldtrap:=trap+4;        (* add 4 to skip over set SR to 2700 *)
X        trap:=ADDRESS(newtrap);
X        CODE(46fcH,2300H);      (* allow interrupts V1.1 *)
X        accsuperstack:=sysvector; (* load value from MX2.ACC *)
X        sysvector:=ADR(sysvar);         (* setup sysvar pointers *)
X        sysvar.currentprocess:=ADR(currentprocess);
X        sysvar.MULTI:=ADR(MULTI);
X        sysvar.slicebegin:=ADR(slicebegin);
X        sysvar.command:=ADR(command);
X        sysvar.request:=ADR(request);
X        sysvar.contextswitch:=ADR(contextswitch);
X        sysvar.CRON:=ADR(CRON);
X        sysvar.spintenable:=ADR(spintenable);
X        sysvar.spintmask:=ADR(spintmask);
X
X        sysvar.spint:=ADR(spint[0]);
X        sysvar.bpsave:=ADR(bpsave);
X        sysvar.pipes:=ADR(pipes);
```

```
X        sysvar.sysmemsize:=ADR(sysmemsize);
X        sysvar.gemsaveGvec:=ADR(gemsaveGvec);
X        sysvar.StartProcess:=StartProcess;
X        sysvar.SwapProcess:=SwapProcess;
X        sysvar.TermProcess:=TermProcess;
X        sysvar.NextPid:=NextPid;
X        sysvar.SleepProcess:=SleepProcess;
X        sysvar.WakeupProcess:=WakeupProcess;
X        sysvar.ChangeProcessPriority:=ChangeProcessPriority;
X        sysvar.MultiBegin:=MultiBegin;
X        sysvar.MultiEnd:=MultiEnd;
X        sysvar.DozeProcess:=DozeProcess;
X        sysvar.WaitProcess:=WaitProcess;
X        sysvar.CronActive:=ADR(CronActive);
X        sysvar.DeviceTable:=ADR(DeviceTable);
X        FOR I:=ORD(dev0) TO ORD(dev7) DO   (* setup user device table *)
X            DeviceTable[I].bconstat:=devstattype(d1);
X            DeviceTable[I].bcostat:=devstattype(d1);
X            DeviceTable[I].bconin:=devintype(d3);
X            DeviceTable[I].bconout:=devouttype(d2);
X        END;
X        FOR I:=0 TO 31 DO       (* clear all pipes *)
X            pipes[I]:=NIL;
X        END;
X        FOR I:=0 TO 15 DO       (* clear all spints *)
X            spint[I].proc:=PROC(NIL);
X        END;
X        slicebegin:=hz200;
X        contextswitch:=0;
X        bios:=biospointer;      (* save original pointer *)
X        oldtermvec:=ptermvec;
X        CODE(4e75H); (* rts *)
X
XEND     settrap;
X(*$P+ *)
X
XPROCEDURE       Initsked;
XVAR             a,b,ssv     :       ADDRESS;
XBEGIN
X        MultiEnd;
X        a:=ADDRESS(OTOS);
X        b:=ADDRESS(MTOS);
X        gemsaveGvec:=a;
X        IF ROMDATE=NEWDATE THEN gemsaveGvec:=b END;
X        SuperExec(SetTimer);
X        NEWPROCESS(sched,ADR(wsp0),TSIZE(workspace),cotick);
X        TRANSFER(x,cotick);
X        schedproc:=currentprocess;
X        SuperExec(settrap);
X        ssv:=0H;
X        Super(ssv);
X        sspval:=ssv;
X        Super(ssv);
```

```
XEND            Initsked;
X
XPROCEDURE      CheckFlag(VAR flag: BOOLEAN): BOOLEAN;
XBEGIN
X       IF flag THEN
X          RETURN TRUE;
X       ELSE
X          RETURN FALSE;
X       END;
XEND    CheckFlag;
X
XPROCEDURE      SetFlag(VAR flag: BOOLEAN);
XBEGIN
X       flag:=TRUE;
XEND    SetFlag;
X
XPROCEDURE      ResetFlag(VAR flag: BOOLEAN);
XBEGIN
X       flag:=FALSE;
XEND    ResetFlag;
X
XPROCEDURE      CheckResetFlag(VAR flag: BOOLEAN): BOOLEAN;
XBEGIN
X       IF flag THEN
X          flag:=FALSE;
X          RETURN TRUE;
X       ELSE
X          RETURN FALSE;
X       END;
XEND    CheckResetFlag;
X
XBEGIN
XEND    ATOMIC.
SHAR_EOF
chmod 0600 ATOMIC.MOD || echo "restore of ATOMIC.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > BITSTUFF.DEF &&
XDEFINITION MODULE BitStuff;
X
XFROM SYSTEM IMPORT BYTE, WORD, LONGWORD;
X
XPROCEDURE LAnd( op1, op2 : LONGWORD):LONGWORD;
XPROCEDURE LOr( op1, op2 : LONGWORD):LONGWORD;
XPROCEDURE LEor( op1, op2 : LONGWORD):LONGWORD;
XPROCEDURE LShl(number : LONGWORD; numbits : CARDINAL):LONGWORD;
XPROCEDURE LShr(number : LONGWORD; numbits : CARDINAL):LONGWORD;
X
XPROCEDURE WAnd( op1, op2 : WORD):WORD;
XPROCEDURE WOr( op1, op2 : WORD):WORD;
XPROCEDURE WEor( op1, op2 : WORD):WORD;
XPROCEDURE WShl(number : WORD; numbits : CARDINAL):WORD;
XPROCEDURE WShr(number : WORD; numbits : CARDINAL):WORD;
X
XPROCEDURE BAnd( op1, op2 : BYTE):BYTE;
```

```
XPROCEDURE BOr( op1, op2 : BYTE):BYTE;
XPROCEDURE BEor( op1, op2 : BYTE):BYTE;
XPROCEDURE BShl(number : BYTE; numbits : CARDINAL):BYTE;
XPROCEDURE BShr(number : BYTE; numbits : CARDINAL):BYTE;
X
XEND BitStuff.
SHAR_EOF
chmod 0600 BITSTUFF.DEF || echo "restore of BITSTUFF.DEF fails"
sed 's/^X//' << 'SHAR_EOF' > BITSTUFF.MOD &&
X(*  Some Functions to perform bit manipulation..  bitsets are a 'bit'
X    clumsy (pun intended) to use and don't operate on LONGWORDS,
X    so i wrote this.
X      It may not be the most efficient, but it is effective, enjoy..
X
X    Russ Damske
X*)
X(*$T-,$S-,$A+ *)
XIMPLEMENTATION MODULE BitStuff;
X
XFROM SYSTEM IMPORT CODE,BYTE,WORD,LONGWORD, SETREG, REGISTER;
X
X(*** FOR LONGWORDS ***)
X
XPROCEDURE LAnd( op1, op2 : LONGWORD):LONGWORD;
X(* logically AND's 2 longwords *)
XBEGIN
X  SETREG( 6, op1 );
X  SETREG( 7, op2 );
X  CODE( 0CE86H );     (* and d6,d7 *)
X  RETURN LONGWORD( REGISTER( 7 ) );
XEND LAnd;
X
XPROCEDURE LOr(op1, op2 : LONGWORD):LONGWORD;
X(* logically OR's 2 longwords *)
XBEGIN
X  SETREG( 6, op1 );
X  SETREG( 7, op2 );
X  CODE( 08E86H );     (* or d6,d7 *)
X  RETURN LONGWORD( REGISTER( 7 ) );
XEND LOr;
X
XPROCEDURE LEor(op1, op2 : LONGWORD):LONGWORD;
X(* logically Exclusive OR of 2 longwords *)
XBEGIN
X  SETREG( 6, op1 );
X  SETREG( 7, op2 );
X  CODE( 0BD87H );     (* eor d6,d7 *)
X  RETURN LONGWORD( REGISTER( 7 ) );
XEND LEor;
X
XPROCEDURE LShl(number : LONGWORD; numbits : CARDINAL):LONGWORD;
X(*  Shifts number an amount (numbits) left *)
XBEGIN
```

```
X  CODE ( 07C00H );    (* moveq.l #0.d6 *)
X  SETREG( 6, numbits );
X  SETREG( 7, number );
X  CODE( 0EDAFH );    (* lsl d6,d7 *)
X  RETURN LONGWORD( REGISTER( 7 ) );
XEND LShl;
X
XPROCEDURE LShr(number : LONGWORD; numbits : CARDINAL):LONGWORD;
X(*  Shifts number an amount (numbits) right *)
XBEGIN
X  CODE( 07C00H );    (* moveq.l #0.d6 *)
X  SETREG( 6, numbits );
X  SETREG( 7, number );
X  CODE( 0ECAFH );    (* lsr d6,d7 *)
X  RETURN LONGWORD( REGISTER( 7 ) );
XEND LShr;
X
X(*** FOR WORDS ***)
X
XPROCEDURE WAnd( op1, op2 : WORD):WORD;
X(* logically AND's 2 words *)
XBEGIN
X  SETREG( 6, op1 );
X  SETREG( 7, op2 );
X  CODE( 0CE46H );     (* and d6,d7 *)
X  RETURN WORD( REGISTER( 7 ) );
XEND WAnd;
X
XPROCEDURE WOr(op1, op2 : WORD):WORD;
X(* logically OR's 2 ords *)
XBEGIN
X  SETREG( 6, op1 );
X  SETREG( 7, op2 );
X  CODE( 08E46H );     (* or d6,d7 *)
X  RETURN WORD( REGISTER( 7 ) );
XEND WOr;
X
XPROCEDURE WEor(op1, op2 : WORD):WORD;
X(* logically Exclusive OR of 2 words *)
XBEGIN
X  SETREG( 6, op1 );
X  SETREG( 7, op2 );
X  CODE( 0BD47H );     (* eor d6,d7 *)
X  RETURN WORD( REGISTER( 7 ) );
XEND WEor;
X
XPROCEDURE WShl(number : WORD; numbits : CARDINAL):WORD;
X(*  Shifts number an amount (numbits) left *)
XBEGIN
X  CODE ( 07C00H );    (* moveq.l #0.d6 *)
X  SETREG( 6, numbits );
X  SETREG( 7, number );
X  CODE( 0ED6FH );    (* lsl d6,d7 *)
```

```
X  RETURN WORD( REGISTER( 7 ) );
XEND WShl;
X
XPROCEDURE WShr(number : WORD; numbits : CARDINAL):WORD;
X(*  Shifts number an amount (numbits) right *)
XBEGIN
X  CODE( 07C00H );    (* moveq.l #0.d6 *)
X  SETREG( 6, numbits );
X  SETREG( 7, number );
X  CODE( 0EC6FH );    (* lsr d6,d7 *)
X  RETURN WORD( REGISTER( 7 ) );
XEND WShr;
X
X(*** FOR BYTES ***)
X
XPROCEDURE BAnd( op1, op2 : BYTE):BYTE;
X(* logically AND's 2 BYTES *)
XBEGIN
X  SETREG( 6, op1 );
X  SETREG( 7, op2 );
X  CODE( 0CE06H );     (* and d6,d7 *)
X  RETURN BYTE( REGISTER( 7 ) );
XEND BAnd;
X
XPROCEDURE BOr(op1, op2 : BYTE):BYTE;
X(* logically OR's 2 BYTES *)
XBEGIN
X  SETREG( 6, op1 );
X  SETREG( 7, op2 );
X  CODE( 08E06H );     (* or d6,d7 *)
X  RETURN BYTE( REGISTER( 7 ) );
XEND BOr;
X
XPROCEDURE BEor(op1, op2 : BYTE):BYTE;
X(* logically Exclusive OR of 2 BYTES *)
XBEGIN
X  SETREG( 6, op1 );
X  SETREG( 7, op2 );
X  CODE( 0BD07H );     (* eor d6,d7 *)
X  RETURN BYTE( REGISTER( 7 ) );
XEND BEor;
X
XPROCEDURE BShl(number : BYTE; numbits : CARDINAL):BYTE;
X(*  Shifts number an amount (numbits) left *)
XBEGIN
X  CODE ( 07C00H );    (* moveq.l #0.d6 *)
X  SETREG( 6, numbits );
X  SETREG( 7, number );
X  CODE( 0ED2FH );    (* lsl d6,d7 *)
X  RETURN BYTE( REGISTER( 7 ) );
XEND BShl;
X
XPROCEDURE BShr(number : BYTE; numbits : CARDINAL):BYTE;
```

```
X(*  Shifts number an amount (numbits) right *)
XBEGIN
X  CODE( 07C00H );    (* moveq.l #0.d6 *)
X  SETREG( 6, numbits );
X  SETREG( 7, number );
X  CODE( 0EC2FH );    (* lsr d6,d7 *)
X  RETURN BYTE( REGISTER( 7 ) );
XEND BShr;
X
XEND BitStuff.
SHAR_EOF
chmod 0600 BITSTUFF.MOD || echo "restore of BITSTUFF.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > CLI.MOD &&
X
X(*              Copyright 1987 fred brooks LogicTek             *)
X(*                                                              *)
X(*                                                              *)
X(*   First Release                    12/8/87-FGB               *)
X(*                                                              *)
X
XMODULE cli;
X(*$T-,$S-,$A+ *)
X(* IMPORT  GEMError; *)
XFROM SYSTEM IMPORT ADDRESS, CODE, REGISTER, SETREG;
XFROM GEMDOS IMPORT Super,Exec,ExecMode,OldTerm;
XFROM BIOS   IMPORT BConStat, BConIn, Device;
XFROM XBIOS  IMPORT SuperExec;
XFROM LOGIN  IMPORT login;
XVAR         cmd,dev,sr,pc,oldbios,sp,tsp,usp,newbios    :       ADDRESS;
X            bios [0b4H]                     :       ADDRESS;
X            d0,ssv,sspval                   :       ADDRESS;
X            i                               :       INTEGER;
X            active                          :       BOOLEAN;
X            delay,result                    :       INTEGER;
X
X(*$P-,$S- *)
XPROCEDURE       multitaskbios;
XBEGIN
X        CODE(2279H,0,4a2H);     (* load saveptr *)
X        CODE(48e1H,1f1fH);      (* save regs *)
X        CODE(23c9H,0,4a2H);     (* update saveptr *)
X        IF active THEN
X           SETREG(8,oldbios);  (* move trap adr *)
X           CODE(43faH,22); (* lea 26(pc),a1 *)
X           CODE(2288H); (* move.l a0,(a1) *)
X           CODE(2279H,0000,04a2H); (* get saveptr *)
X           CODE(4cd9H,0f8f8H); (* restore regs movem *)
X           CODE(23c9H,0000,04a2H); (* update saveptr *)
X           CODE(4ef9H,0,0) (* jmp back to routine *)
X        END;
X        CODE(306fH,0);         (* move.w 0(a7),a0  get sr *)
X        sr:=REGISTER(8);
X        CODE(206fH,2);         (* move.l 2(a7),a0  get pc *)
```

```
X       pc:=REGISTER(8);
X
X       IF sr<3ffH THEN        (* called from user mode *)
X          CODE(4e69H);        (* move.l usp,a1 *)
X          CODE(3069H,2);      (* move.w 2(a1),a0 *)
X          dev:=REGISTER(8);
X          CODE(4e69H);        (* move.l usp,a1 *)
X          CODE(3069H,0);      (* move.w 0(a1),a0 *)
X          cmd:=REGISTER(8);
X          IF (cmd=2) THEN
X                active:=TRUE;
X                INC(delay);
X                CODE(5c8fH); (* addq.l #6,a7 correct stack *)
X                sp:=REGISTER(15);
X                REPEAT
X                   (* set user mode then back to super *)
X                   IF sp=sspval THEN
X                      tsp:=REGISTER(15);
X                      CODE(4e68H);     (* move.l usp,a0 *)
X                      usp:=REGISTER(8);
X                      CODE(46fcH,300H); (* move.w $300,sr *)
X                      FOR i:=0 TO 10 DO END; (* busy loop for mx2 *)
X                      CODE(42a7H);      (* clr.l  -(sp) *)
X                      CODE(3f3cH,20H);  (* move.w $20,-(sp) *)
X                      CODE(4e41H);      (* trap #1 *)
X                      SETREG(15,tsp);
X                      SETREG(8,usp);
X                      CODE(4e60H);     (* move.l a0,usp *)
X                   END;
X                   CODE(48e7H,7ffeH); (* save regs *)
X                   CODE(9bcdH);       (* clr a5 *)
X                   newbios:=bios;
X                   bios:=oldbios;
X                   SETREG(0,0);
X                   IF BConStat(Device(dev)) THEN SETREG(0,-1) END;
X                   bios:=newbios;
X                   CODE(4cdfH,7ffeH); (* restore regs *)
X                   d0:=REGISTER(0);
X                UNTIL d0#0;
X                CODE(48e7H,7ffeH); (* save regs *)
X                CODE(9bcdH);       (* clr a5 *)
X                newbios:=bios;
X                bios:=oldbios;
X                SETREG(0,BConIn(Device(dev)));
X                bios:=newbios;
X                CODE(4cdfH,7ffeH); (* restore regs *)
X                d0:=REGISTER(0);
X                SETREG(9,pc);
X                CODE(2f09H);        (* push pc to stack *)
X                SETREG(9,sr);
X                CODE(3f09H);        (* push sr to stack *)
X                active:=FALSE;
X                SETREG(0,d0);       (* move char to d0 *)
```

```
X                CODE(2279H,0000,04a2H); (* get saveptr *)
X                CODE(4cd9H,0f8f8H); (* restore regs movem *)
X                CODE(23c9H,0000,04a2H); (* update saveptr *)
X                CODE(4e73H);        (* rte *)
X           END;
X           IF (cmd=1) THEN
X                active:=TRUE;
X                INC(delay);
X                CODE(5c8fH); (* addq.l #6,a7 correct stack *)
X                sp:=REGISTER(15);
X                CODE(48e7H,7ffeH); (* save regs *)
X                CODE(9bcdH);       (* clr a5 *)
X                newbios:=bios;
X                bios:=oldbios;
X                SETREG(0,0);
X                IF BConStat(Device(dev)) THEN SETREG(0,-1) END;
X                bios:=newbios;
X                d0:=REGISTER(0);
X                CODE(4cdfH,7ffeH); (* restore regs *)
X                IF d0=0 THEN
X                   (* set user mode then back to super *)
X                   IF sp=sspval THEN
X                      tsp:=REGISTER(15);
X                      CODE(4e68H);     (* move.l usp,a0 *)
X                      usp:=REGISTER(8);
X                      CODE(46fcH,300H); (* move.w $300,sr *)
X                      FOR i:=0 TO 10 DO END; (* busy loop for mx2 *)
X                      CODE(42a7H);      (* clr.l  -(sp) *)
X                      CODE(3f3cH,20H);  (* move.w $20,-(sp) *)
X                      CODE(4e41H);      (* trap #1 *)
X                      SETREG(15,tsp);
X                      SETREG(8,usp);
X                      CODE(4e60H);     (* move.l a0,usp *)
X                   END;
X                END;
X                SETREG(9,pc);
X                CODE(2f09H);        (* push pc to stack *)
X                SETREG(9,sr);
X                CODE(3f09H);        (* push sr to stack *)
X                active:=FALSE;
X                SETREG(0,d0);       (* move char to d0 *)
X                CODE(2279H,0000,04a2H); (* get saveptr *)
X                CODE(4cd9H,0f8f8H); (* restore regs movem *)
X                CODE(23c9H,0000,04a2H); (* update saveptr *)
X                CODE(4e73H);        (* rte *)
X           END;
X        ELSE                       (* called from super mode *)
X           CODE(306fH,8);  (* move.w 8(a7),a0 *)
X           dev:=REGISTER(8);
X           CODE(306fH,6);  (* move.w 6(a7),a0 *)
X           cmd:=REGISTER(8);
X           IF (cmd=2) THEN
X                CODE(5c8fH); (* addq.l #6,a7 correct stack *)
```

```
X                REPEAT
X                   CODE(48e7H,7ffeH); (* save regs *)
X                   CODE(9bcdH);       (* clr a5 *)
X                   newbios:=bios;
X                   bios:=oldbios;
X                   SETREG(0,0);
X                   IF BConStat(Device(dev)) THEN SETREG(0,-1) END;
X                   bios:=newbios;
X                   CODE(4cdfH,7ffeH); (* restore regs *)
X                   d0:=REGISTER(0);
X                UNTIL d0#0;
X                CODE(48e7H,7ffeH); (* save regs *)
X                CODE(9bcdH);       (* clr a5 *)
X                newbios:=bios;
X                bios:=oldbios;
X                SETREG(0,BConIn(Device(dev)));
X                bios:=newbios;
X                CODE(4cdfH,7ffeH); (* restore regs *)
X                d0:=REGISTER(0);
X                SETREG(9,pc);
X                CODE(2f09H);        (* push pc to stack *)
X                SETREG(9,sr);
X                CODE(3f09H);        (* push sr to stack *)
X                SETREG(0,d0);       (* move char to d0 *)
X                CODE(2279H,0000,04a2H); (* get saveptr *)
X                CODE(4cd9H,0f8f8H); (* restore regs movem *)
X                CODE(23c9H,0000,04a2H); (* update saveptr *)
X                CODE(4e73H);        (* rte *)
X             END;
X          END;
X
X          SETREG(8,oldbios);  (* move trap adr *)
X          CODE(43faH,22); (* lea 26(pc),a1 *)
X          CODE(2288H); (* move.l a0,(a1) *)
X          CODE(2279H,0000,04a2H); (* get saveptr *)
X          CODE(4cd9H,0f8f8H); (* restore regs movem *)
X          CODE(23c9H,0000,04a2H); (* update saveptr *)
X          CODE(4ef9H,0,0) (* jmp back to routine *)
X
XEND               multitaskbios;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE         dummyvector;
XBEGIN
X          CODE(4e73H); (* rte *)
XEND               dummyvector;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE         setup;
XBEGIN
X          oldbios:=bios;
```

```
X        bios:=ADDRESS(multitaskbios);
X        CODE(4e75H);    (* RTS *)
XEND             setup;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE       setdown;
XBEGIN
X        bios:=oldbios;
X        CODE(4e75H);    (* RTS *)
XEND             setdown;
X(*$P+ *)
X
XBEGIN;
X    ssv:=0H;
X    Super(ssv);
X    sspval:=ssv;
X    Super(ssv);
X    SuperExec(setup);
X    login;
X    SuperExec(setdown);
X    OldTerm;
XEND cli.
SHAR_EOF
chmod 0600 CLI.MOD || echo "restore of CLI.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > COM.MOD &&
X
X(*              Copyright 1987 fred brooks LogicTek             *)
X(*                                                              *)
X(*                                                              *)
X(*   First Release                      12/8/87-FGB             *)
X(*                                                              *)
X
XMODULE  com;
XFROM    XBIOS   IMPORT  ConfigureRS232,SerialSpeed,FlowFlavor;
XFROM    BIOS    IMPORT  BConStat,BCosStat,BConIn,BConOut,Device;
XFROM    GEMX    IMPORT  BasePageAddress;
XFROM    GEMDOS  IMPORT  OldTerm;
XVAR     longchar        :       LONGCARD;
X        t               :       BITSET;
X        char            :       CHAR;
X        port            :       Device;
XBEGIN
X        char:=BasePageAddress^.CmdLine[1];
X        IF (char='m') OR (char='M') THEN
X           port:=HSS;
X        ELSE
X           port:=AUX;
X        END;
X        IF port=AUX THEN
X           IF (char=0c) THEN
X              ConfigureRS232(BPS1200,NONE,-1,-1,-1,-1);
X           END;
```

```
X           IF (char='0') THEN
X              ConfigureRS232(BPS300,NONE,-1,-1,-1,-1);
X           END;
X           IF (char='1') THEN
X              ConfigureRS232(BPS1200,NONE,-1,-1,-1,-1);
X           END;
X           IF (char='2') THEN
X              ConfigureRS232(BPS2400,NONE,-1,-1,-1,-1);
X           END;
X           IF (char='4') THEN
X              ConfigureRS232(BPS4800,NONE,-1,-1,-1,-1);
X           END;
X           IF (char='9') THEN
X              ConfigureRS232(BPS9600,NONE,-1,-1,-1,-1);
X           END;
X           IF (char='h') OR (char='H') THEN
X              ConfigureRS232(BPS19200,NONE,-1,-1,-1,-1);
X           END;
X        END;
X        LOOP
X                IF BConStat(CON) THEN (* read keyboard *)
X                   longchar:=BConIn(CON);
X                   IF (longchar DIV 65536) = 61H THEN EXIT END;
X                   char:=CHAR(longchar);
X                   BConOut(port,char);
X                END;
X
X                IF BConStat(port) THEN (* read com port *)
X                   longchar:=BConIn(port);
X                   t:=BITSET(longchar);
X                   EXCL(t,7);
X                   char:=CHAR(t);
X                   BConOut(CON,char);
X                END;
X        END;
X        OldTerm;
XEND     com.
SHAR_EOF
chmod 0600 COM.MOD || echo "restore of COM.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > KILLLP.MOD &&
XMODULE  killlp;
XFROM    SYSCALL         IMPORT  Trigger,SpintInfo;
XFROM    GEMDOS          IMPORT  Term;
XFROM    SYSTEM          IMPORT  ADDRESS,ADR;
XFROM    Terminal        IMPORT  WriteString,WriteLn;
XVAR     ok              :       BOOLEAN;
X        spintcmd        :       POINTER TO ARRAY [0..1] OF LONGCARD;
XBEGIN
X        IF SpintInfo(0,spintcmd) THEN
X           spintcmd^[0]:=99;
X           ok:=Trigger(0);
X        ELSE
X           WriteLn;
```

```
X           WriteString("Background spooler not running.");
X           WriteLn;
X           ok:=Term(-1);
X        END;
X        ok:=Term(0);
XEND     killlp.
SHAR_EOF
chmod 0600 KILLLP.MOD || echo "restore of KILLLP.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > LOGIN.DEF &&
XDEFINITION MODULE LOGIN;
X
X(* read passwd file and sign on user *)
XPROCEDURE       login;
X
X(* encrypt password *)
XPROCEDURE       crypt(VAR red,crypto: ARRAY OF CHAR);
X
XEND     LOGIN.
X
SHAR_EOF
chmod 0600 LOGIN.DEF || echo "restore of LOGIN.DEF fails"
sed 's/^X//' << 'SHAR_EOF' > LOGIN.MOD &&
XIMPLEMENTATION MODULE  LOGIN;
SHAR_EOF
echo "End of part 2, continue with part 3"
echo "3" > s2_seq_.tmp
exit 0


From -3801903554389291715
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,f725dfdcacd2bc80,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-12-27 00:08:02 PST
Path: utzoo!attcan!uunet!husc6!bloom-beacon!tut.cis.ohio-state.edu!rutgers!ucsd!
ucbvax!agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i083: mx2v230 -- Multitasking kernel with utilities  part03/08
Keywords: shar, Modula-2
Message-ID: <5867@saturn.ucsc.edu>
Date: 27 Dec 88 08:08:02 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 920
Approved: koreth@ssyx.ucsc.edu

Submitted-by: madsen@sask.usask.ca (Jorgen Madsen)
Posting-number: Volume 1, Issue 83
Archive-name: mx2v230/part03

#!/bin/sh
# this is part 3 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
```

```
# file LOGIN.MOD continued
#
CurArch=3
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> LOGIN.MOD
XFROM    Terminal   IMPORT  ReadString,WriteString,WriteLn;
XFROM    TermBase   IMPORT  DoWrite,WriteProc;
XFROM    Strings    IMPORT  String,Compare,CompareResults,Pos,Copy,Concat,
X                           Length;
XFROM    GEMDOS     IMPORT  OldTerm,SetPath,GetPath,Exec,ExecMode,SetDrv;
XIMPORT  TextIO;
XFROM    Streams    IMPORT  StreamKinds,CloseStream;
XFROM    SYSTEM     IMPORT  ADR;
XCONST   USERS = 16;
X        SEED1 = 3844364602;
X        SEED2 = 1561798242;
XTYPE    sarray  =       ARRAY [0..7] OF CHAR;
XVAR     user,password,prg       :       ARRAY [0..USERS-1] OF String;
X        path,com                :       ARRAY [0..USERS-1] OF String;
X        input                   :       String;
X        black,gulampath         :       String;
X        clipath                 :       String;
X        i,j,index,spos,epos     :       CARDINAL;
X        userfound,ok            :       BOOLEAN;
X        normal,noecho           :       WriteProc;
X        result                  :       INTEGER;
X        drvmap                  :       LONGCARD;
X
XPROCEDURE       NoEcho(char: CHAR);
XBEGIN
XEND             NoEcho;
X
XPROCEDURE       login;
XBEGIN
X        TextIO.SetDefaultIO("PASSWD",READ,result);
X        normal:=DoWrite;
X        noecho:=NoEcho;
X        FOR i:=0 TO USERS-1 DO               (* read in userfile *)
X                TextIO.ReadString(input);
X                spos:=0;
X                ok:=Pos(input,":",spos,epos);
X                Copy(input,spos,epos-spos,user[i]);
X                spos:=epos+1;
X                ok:=Pos(input,":",spos,epos);
X                Copy(input,spos,epos-spos,password[i]);
```

```
X               spos:=epos+1;
X               ok:=Pos(input,":",spos,epos);
X               Copy(input,spos,epos-spos,path[i]);
X               spos:=epos+1;
X               ok:=Pos(input,":",spos,epos);
X               Copy(input,spos,epos-spos,prg[i]);
X               spos:=epos+1;
X               Copy(input,spos,Length(input)-spos,com[i]);
X         END; (* for *)
X         CloseStream(TextIO.in,result);
X         CloseStream(TextIO.out,result);
X         input[0]:=33c; input[1]:='e'; input[2]:=0c;
X         WriteString(input);
X         LOOP
X            REPEAT
X               userfound:=FALSE;
X               WriteLn;
X               WriteString("Login: ");
X               ReadString(input);
X               WriteLn;
X            UNTIL input[0]#0c;
X            FOR i:=0 TO USERS-1 DO
X                IF Compare(input,user[i])=Equal THEN
X                   userfound:=TRUE;
X                   index:=i;
X                END;
X            END;
X            IF (password[index][0]#0c) OR (NOT userfound) THEN
X            WriteString("Password: ");
X            DoWrite:=noecho;
X            ReadString(input);
X            DoWrite:=normal;
X            WriteLn;
X            black:=user[index];
X            crypt(input,black);
X            IF (Compare(black,password[index])=Equal) AND userfound THEN
X               EXIT;
X            END;
X            END;
X            IF (password[index][0]=0c) AND userfound THEN EXIT END;
X            WriteLn;
X            WriteString("Login incorrect");
X         END; (* loop *)
X         GetPath(gulampath,0);
X         IF path[index][0]=0c THEN
X            GetPath(clipath,0);
X         ELSE
X            clipath:=path[index];
X            IF clipath[0]#'\' THEN
X               IF ORD(clipath[0])>60H THEN
X                  SetDrv(ORD(clipath[0])-61H,drvmap);
X               ELSE
X                  SetDrv(ORD(clipath[0])-41H,drvmap);
```

```
X              END;
X              j:=1;
X              REPEAT
X                clipath[j-1]:=clipath[j];
X                INC(j);
X              UNTIL (clipath[j-1]=0c);
X           END;
X        END;
X        IF NOT ((clipath[0]='\') AND (clipath[1]=0c)) THEN
X           Concat(clipath,"\",clipath);
X        END;
X        IF prg[index][0]=0c THEN
X           prg[index]:=gulampath;
X        IF NOT ((prg[index][0]='\') AND (prg[index][1]=0c)) THEN
X           Concat(prg[index],"\",prg[index]);
X        END;
X           Concat(prg[index],"gulam.prg",prg[index]);
X        END;
X        ok:=SetPath(clipath);
X        input[0]:=33c; input[1]:='f'; input[2]:=0c;
X        WriteString(input);
X        Exec(loadExecute,prg[index],"","",result);
XEND     login;
X
XPROCEDURE       crypt(VAR red,crypto: ARRAY OF CHAR);
XVAR     seed            :       ARRAY [0..1] OF LONGCARD;
X        chptr           :       POINTER TO sarray;
X        black           :       CHAR;
XBEGIN
X        seed[0]:=SEED1;
X        seed[1]:=SEED2;
X        chptr:=ADR(seed);
X        FOR i:=0 TO 7 DO
X            FOR j:=0 TO 7 DO
X                chptr^[i]:=CHAR(BITSET(chptr^[i])/BITSET(red[j]));
X                chptr^[i]:=CHAR(BITSET(chptr^[i])/BITSET(crypto[j]));
X            END;
X            black:=CHAR(40H+CARDINAL( BITSET(chptr^[i]) * BITSET(31) ));
X            crypto[i]:=black;
X        END;
X        crypto[8]:=0c;
XEND             crypt;
X
XBEGIN
XEND     LOGIN.
X
SHAR_EOF
chmod 0600 LOGIN.MOD || echo "restore of LOGIN.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > LP.MOD &&
XMODULE  lp;
XFROM    SYSCALL         IMPORT  Trigger,SpintInfo;
XFROM    GEMDOS          IMPORT  Term;
XFROM    SYSTEM          IMPORT  ADDRESS,ADR;
```

```
XFROM    Terminal        IMPORT  WriteString,WriteLn;
XVAR     ok              :       BOOLEAN;
X        spintcmd        :       POINTER TO ARRAY [0..1] OF LONGCARD;
XBEGIN
X        IF SpintInfo(0,spintcmd) THEN
X           spintcmd^[0]:=1;
X           ok:=Trigger(0);
X        ELSE
X           WriteLn;
X           WriteString("Background spooler not running.");
X           WriteLn;
X           ok:=Term(-1);
X        END;
X        ok:=Term(0);
XEND     lp.
SHAR_EOF
chmod 0600 LP.MOD || echo "restore of LP.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > MX2.MOD &&
X(*$T-,$S-,$A+ *)
XMODULE MX2;
X
X(*              Copyright 1987,1988 fred brooks LogicTek        *)
X(*                                                              *)
X(*                                                              *)
X(*   First Release                      12/8/87-FGB             *)
X(* Corrected code to match changes in lib modules               *)
X(*                                      1/9/88-FGB              *)
X(* Bug in parser converted all text to UPPER case. Fixed        *)
X(*                                      2/27/88-FGB             *)
X(* Misc bug fixes.                                              *)
X(*                                      4/3/88-FGB              *)
X(* Remove NETWORK routines from kernel                          *)
X(*                                      4/11/88-FGB             *)
X(* Remove SP & XMODEM  routines from kernel                     *)
X(*                                      4/11/88-FGB             *)
X(* Remove TRAP15 interface routines     6/9/88-FGB              *)
X(*                                                              *)
X
X  FROM Terminal       IMPORT     ReadString,WriteString,WriteLn;
X  FROM TextIO         IMPORT     REadString;
X  FROM Conversions    IMPORT     ConvertFromString;
X  FROM M2Conversions  IMPORT     ConvertToInteger,ConvertToAddr;
X  FROM BitStuff       IMPORT     WAnd,WShr;
X  FROM GEMDOS         IMPORT     ExecMode,Exec,Alloc,Free,OldTerm,
X                                 GetPath,GetDrv,GetTime,
X                                 SetPath,SetDrv;
X  FROM XBIOS          IMPORT     SuperExec,IOREC,IORECPTR,SerialDevice,
X                                 IORec,ScreenPhysicalBase;
X  FROM BIOS           IMPORT     Device,BConStat,BConIn,BConOut,BCosStat,
X                                 KBShifts,GetKBShift,KBShiftBits;
X  FROM Streams        IMPORT     Stream,OpenStream,CloseStream,EOS,
X                                 StreamKinds;
X  FROM Storage        IMPORT     CreateHeap;
```

```
X  FROM SYSTEM         IMPORT     ADR,ADDRESS,CODE,PROCESS,REGISTER,SETREG;
X
X  FROM   ATOMIC   IMPORT  Initsked,MultiEnd,MultiBegin,CronActive,
X                          InitProcesses,StartProcess,currentprocess,
X                          TermProcess,SIGNAL,SwapProcess,request,MAGIC,
X                          command,SleepProcess,WakeupProcess,
X                          ChangeProcessPriority,CRON,DeviceTable,
X                          spintenable,spintmask,spint,bpsave,GEMTYPE,
X                          sysvariable,gemsaveGvec,ROMDATE,OLDDATE,NEWDATE,
X                          NextPid,VERSION,sysmemsize,devicetype;
X
X  FROM   SCANNER  IMPORT  scinit,nxparm,ltext,etext,bkparm,state;
X
X  FROM   Strings  IMPORT  Compare,Pos,Length,Concat,CompareResults,String;
X
X
XCONST   intnum          =       4;    (* interrupt number on MFP *)
XTYPE    ctype   =
X                  RECORD
X                        stime            :       LONGCARD;
X                        freq             :       LONGCARD;
X                        btime            :       LONGCARD;
X                        command          :       String;
X                        active           :       BOOLEAN;
X                  END;
X          screen  =     ARRAY [0..7999] OF LONGCARD;
XVAR
X       result,pri,cli1,cli2,clipid,
X       spawnpid                                    : INTEGER;
X       proc                                        : PROC;
X       Oportdevice,Iportdevice                     : devicetype;
X       pc,returnadr,kpc,
X       oldikbd,par                                 : ADDRESS;
X       gemsave,param                               : ARRAY [0..15] OF ADDRESS;
X       paramstringptr                              : POINTER TO String;
X       sizewsp,temphz200,cronslice,currenttime     : LONGCARD;
X       cmd,dev,c,a7,SR,tbiosSave                   : ADDRESS;
X       gem   [88H]                                 : ADDRESS;
X       hz200  [4baH]                               : LONGCARD;
X       termvec [408H]                              : ADDRESS;
X       linea [28H]                                 : ADDRESS;
X       gemdos [84H]                                : ADDRESS;
X       gsxgem [88H]                                : ADDRESS;
X       tbios  [0b4H]                               : ADDRESS;
X       xbios  [0b8H]                               : ADDRESS;
X       linef  [2cH]                                : ADDRESS;
X       level2 [68H]                                : ADDRESS;
X       level4 [70H]                                : ADDRESS;
X       shellp [04f6H]                              : ADDRESS;
X       ikbdvec [118H]                              : PROC;
X       OpenCLI,i,bprunning,function,
X       time,defaultdrv,requestdrv,sr,drv,
X       ksr                                         : CARDINAL;
```

```
X       cmdstring,temp,name,tail,envstr,pname,
X       defaultpath,requestpath,pstemp,initprg   : String;
X       inuse,done,
X       swloaded,caps,reservemem,swapcli,inok,
X       outok                                    : BOOLEAN;
X       periods,drivemap,HotKey,Hotreturn,kjunk,
X       NorMouse,CurMouse,RebootKey,memreserve,
X       SYSMEM,cin                               : LONGCARD;
X       crontable                                : ARRAY [0..15] OF ctype;
X       ticktime                                 : LONGINT;
X       s0                                       : SIGNAL;
X       sysvar                                   : sysvariable;
X       sysvector [144H]                         : POINTER TO sysvariable;
X       Kshift,Hotset,CapsL                      : KBShifts;
X       physcreen                                : POINTER TO screen;
X       screensave                               : POINTER TO ARRAY [1..2]
X                                                  OF screen;
X       kbdiorec                                 : IORECPTR;
X       ibuf                                     : POINTER TO ARRAY [0..63]
X                                                  OF LONGCARD;
X
XCONST
X  TDI   = "                Written in TDI MODULA-2 Version 3.01a ";
X  TITLE1 = "       ";
X  TITLE2 = " Copyright LogicTek 1987,1988 Fred Brooks ";
X  CRONFILE = "CRONTAB";
X
X(*$P- *)
XPROCEDURE       keytrapstart; (* modify IKBD system vector *)
XBEGIN
X        CODE(48e7H,0fffeH);   (* save regs movem  *)
X        CODE(206fH,62); (* move.l 62(a7),a0 get pc *)
X        kpc:=REGISTER(8);
X        CODE(306fH,60); (* move.w 60(a7),a0 get sr *)
X        ksr:=CARDINAL(REGISTER(8));
X        SETREG(8,ADDRESS(keytrapend));
X        CODE(2f48H,62); (* move new pc to stack *)
X        SETREG(8,2700H);
X        CODE(3f48H,60); (* move new sr to stack *)
X
X        SETREG(8,oldikbd);  (* move IKBD trap adr *)
X        CODE(43faH,10); (* lea 12(pc),a1 *)
X        CODE(2288H); (* move.l a0,(a1) *)
X        CODE(4cdfH,7fffH); (* restore regs movem *)
X        CODE(4ef9H,0,0) (* jmp back to routine *)
XEND     keytrapstart;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE       keytrapend; (* check for hotkeys *)
XBEGIN
X        CODE(5d8fH);    (* subq.l #6,sp *)
X        CODE(48e7H,0fffeH);   (* save regs movem  *)
```

```
X
X        Hotreturn:=ibuf^[kbdiorec^.ibuftl DIV 4];
X        CODE(2f39H,0,4a2H); (* save BIOS pointers *)
X        CODE(4b9H,0,2eH,0,4a2H);
X
X        IF Hotreturn=RebootKey THEN
X           CODE(46fcH,0300H); (* set user mode *)
X           CODE(42a7H,3f3cH,20H,4e41H,42b9H,0H,420H,2079H,0H,4H,4ed0H);
X        END;
X        IF Hotreturn=NorMouse THEN
X           gkey;
X           BConOut(KDB,CHAR(08H)); (* send relative mouse *)
X        END;
X        IF Hotreturn=CurMouse THEN
X           gkey;
X           BConOut(KDB,CHAR(0aH)); (* send cursor mouse *)
X        END;
X        IF Hotreturn=HotKey THEN
X           gkey;
X           swapcli:=TRUE;
X        END;
X
X        CODE(23dfH,0,4a2H); (* restore BIOS pointers *)
X
X        SETREG(8,ADDRESS(kpc));
X        CODE(2f48H,62); (* move new pc to stack *)
X        SETREG(8,ADDRESS(ksr));
X        CODE(3f48H,60); (* move new sr to stack *)
X        CODE(4cdfH,7fffH); (* restore regs movem *)
X        CODE(4e73H); (* rte *)
XEND     keytrapend;
X(*$P- *)
X
X(*$P- *)
XPROCEDURE       gkey;
XBEGIN
X        IF BConStat(CON) THEN
X           kjunk:=BConIn(CON);
X           ibuf^[kbdiorec^.ibuftl DIV 4]:=0;
X        END;
X        CODE(4e75H); (* rts *)
XEND             gkey;
X(*$P+ *)
X
XPROCEDURE       SetDrvPath(drive: CARDINAL; VAR path: ARRAY OF CHAR);
XBEGIN
X        SetDrv(drive,drivemap);
X        IF path[0]=0c THEN
X           path[0]:='\';
X           path[1]:=0c;
X           path[2]:=0c;
X        END;
X        done:=SetPath(path);
```

```
XEND            SetDrvPath;
X
X(* this is for the BIOS devices *)
X(*$P-,$S- *)
XPROCEDURE      changedevbios;
XBEGIN
X       CODE(48e7H,07ffeH);     (* save regs *)
X       CODE(306fH,56);         (* move.w 56(a7),a0  get SR *)
X       SR:=REGISTER(8);
X       IF SR<3ffH THEN        (* called from user mode *)
X          CODE(204fH);            (* move.l a7,a0 *)
X          a7:=REGISTER(8);        (* save ssp *)
X          CODE(4e68H,2e48H);      (* move.l usp,a0  move.l a0,a7 *)
X          CODE(306fH,2);  (* move.w 2(a7),a0 *)
X          dev:=REGISTER(8);
X          CODE(306fH,0);  (* move.w 0(a7),a0 *)
X          cmd:=REGISTER(8);
X          IF (cmd=1) OR (cmd=2) THEN (* check INPUT *)
X             IF dev=2 THEN
X                IF currentprocess^.Iport=null THEN
X                   SETREG(8,a7);
X                   CODE(2e48H);            (* move.l a0,a7 *)
X                   CODE(4cdfH,7ffeH);         (* restore regs movem *)
X                   CODE(203CH,0FFFFH,0FFFFH); (* move.l -1,d0 *)
X                   CODE(4E73H);  (* RTE return -1 on all calls *)
X                END;
X                IF ORD(currentprocess^.Iport)>ORD(null) THEN (* user *)
X                    IF cmd=2 THEN
X                cin:=DeviceTable[ORD(currentprocess^.Oport)].bconin();
X                      SETREG(8,a7);
X                      CODE(2e48H);            (* move.l a0,a7 *)
X                      SETREG(0,ADDRESS(cin));
X                      CODE(4cdfH,7ffeH);         (* restore regs movem *)
X                      CODE(4E73H);  (* RTE return -1 on all calls *)
X                   ELSE
X                inok:=DeviceTable[ORD(currentprocess^.Oport)].bconstat();
X                      IF inok THEN
X                         SETREG(8,a7);
X                         CODE(2e48H);            (* move.l a0,a7 *)
X                         CODE(4cdfH,7ffeH);      (* restore regs movem *)
X                         CODE(203CH,0FFFFH,0FFFFH); (* move.l -1,d0 *)
X                         CODE(4E73H);  (* RTE return -1 *)
X                      ELSE
X                         SETREG(8,a7);
X                         CODE(2e48H);            (* move.l a0,a7 *)
X                         CODE(4cdfH,7ffeH);      (* restore regs movem *)
X                         CODE(203CH,0H,0H); (* move.l 0,d0 *)
X                         CODE(4E73H);  (* RTE return 0 *)
X                      END;
X                   END;
X               END;
X                dev:=ADDRESS(currentprocess^.Iport); (* change to port *)
X                SETREG(8,dev);
```

```
X                CODE(3f48H,2); (* move.w a0,2(a7) set value in stack *)
X             END;
X          END;
X          IF (cmd=3) OR (cmd=8) THEN (* check OUTPUT *)
X             IF dev=2 THEN
X                IF currentprocess^.Oport=null THEN
X                   SETREG(8,a7);
X                   CODE(2e48H);            (* move.l a0,a7 *)
X                   CODE(4cdfH,7ffeH);         (* restore regs movem *)
X                   CODE(203CH,0FFFFH,0FFFFH); (* move.l -1,d0 *)
X                   CODE(4E73H);  (* RTE return -1 on all calls *)
X                END;
X                IF ORD(currentprocess^.Oport)>ORD(null) THEN
X                   IF cmd=3 THEN
X                      CODE(306fH,4);  (* move.w 4(a7),a0 *)
X                      c:=REGISTER(8);
X                DeviceTable[ORD(currentprocess^.Oport)].bconout(CHAR(c));
X                      SETREG(8,a7);
X                      CODE(2e48H);            (* move.l a0,a7 *)
X                      CODE(4cdfH,7ffeH);         (* restore regs movem *)
X                      CODE(203CH,0FFFFH,0FFFFH); (* move.l -1,d0 *)
X                      CODE(4E73H);  (* RTE return -1 on all calls *)
X                   ELSE
X                outok:=DeviceTable[ORD(currentprocess^.Oport)].bcostat();
X                      IF outok THEN
X                         SETREG(8,a7);
X                         CODE(2e48H);            (* move.l a0,a7 *)
X                         CODE(4cdfH,7ffeH);      (* restore regs movem *)
X                         CODE(203CH,0FFFFH,0FFFFH); (* move.l -1,d0 *)
X                         CODE(4E73H);  (* RTE return -1 *)
X                      ELSE
X                         SETREG(8,a7);
X                         CODE(2e48H);            (* move.l a0,a7 *)
X                         CODE(4cdfH,7ffeH);      (* restore regs movem *)
X                         CODE(203CH,0H,0H); (* move.l 0,d0 *)
X                         CODE(4E73H);  (* RTE return 0 *)
X                      END;
X                   END;
X                END;
X                dev:=ADDRESS(currentprocess^.Oport); (* change to port *)
X                SETREG(8,dev);
X                CODE(3f48H,2); (* move.w a0,2(a7) set value in stack *)
X             END;
X          END;
X          SETREG(8,a7);
X          CODE(2e48H);            (* move.l a0,a7 *)
X       ELSE                       (* called from super mode *)
X          CODE(306fH,64);  (* move.w 64(a7),a0 *)
X          dev:=REGISTER(8);
X          CODE(306fH,62);  (* move.w 62(a7),a0 *)
X          cmd:=REGISTER(8);
X          IF (cmd=1) OR (cmd=2) THEN (* check INPUT *)
X             IF dev=2 THEN
```

```
X                IF currentprocess^.Iport=null THEN
X                   CODE(4cdfH,7ffeH);         (* restore regs movem *)
X                   CODE(203CH,0FFFFH,0FFFFH); (* move.l -1,d0 *)
X                   CODE(4E73H); (* RTE return -1 on all calls *)
X                END;
X                IF ORD(currentprocess^.Iport)>ORD(null) THEN
X                    IF cmd=2 THEN
X                cin:=DeviceTable[ORD(currentprocess^.Oport)].bconin();
X                      SETREG(0,ADDRESS(cin));
X                      CODE(4cdfH,7ffeH);         (* restore regs movem *)
X                      CODE(4E73H);  (* RTE return cin *)
X                   ELSE
X                inok:=DeviceTable[ORD(currentprocess^.Oport)].bconstat();
X                      IF inok THEN
X                         CODE(4cdfH,7ffeH);   (* restore regs movem *)
X                         CODE(203CH,0FFFFH,0FFFFH); (* move.l -1,d0 *)
X                         CODE(4E73H);  (* RTE return -1  *)
X                      ELSE
X                         CODE(4cdfH,7ffeH);   (* restore regs movem *)
X                         CODE(203CH,0H,0H); (* move.l 0,d0 *)
X                         CODE(4E73H);  (* RTE return 0 *)
X                      END;
X                   END;
X                 END;
X                  dev:=ADDRESS(currentprocess^.Iport); (* change to port *)
X                  SETREG(8,dev);
X                  CODE(3f48H,64); (* move.w a0,64(a7) set value in stack *)
X             END;
X          END;
X          IF (cmd=3) OR (cmd=8) THEN (* check OUTPUT *)
X             IF dev=2 THEN
X                IF currentprocess^.Oport=null THEN
X                   CODE(4cdfH,7ffeH);         (* restore regs movem *)
X                   CODE(203CH,0FFFFH,0FFFFH); (* move.l -1,d0 *)
X                   CODE(4E73H); (* RTE return -1 on all calls *)
X                END;
X                IF ORD(currentprocess^.Oport)>ORD(null) THEN
X                   IF cmd=3 THEN
X                      CODE(306fH,66);  (* move.w 66(a7),a0 *)
X                      c:=REGISTER(8);
X                DeviceTable[ORD(currentprocess^.Oport)].bconout(CHAR(c));
X                      CODE(4cdfH,7ffeH);         (* restore regs movem *)
X                      CODE(203CH,0FFFFH,0FFFFH); (* move.l -1,d0 *)
X                      CODE(4E73H);  (* RTE return -1 on all calls *)
X                   ELSE
X                outok:=DeviceTable[ORD(currentprocess^.Oport)].bcostat();
X                      IF outok THEN
X                         CODE(4cdfH,7ffeH);   (* restore regs movem *)
X                         CODE(203CH,0FFFFH,0FFFFH); (* move.l -1,d0 *)
X                         CODE(4E73H);  (* RTE return -1  *)
X                      ELSE
X                         CODE(4cdfH,7ffeH);   (* restore regs movem *)
X                         CODE(203CH,0H,0H); (* move.l 0,d0 *)
```

```
X                          CODE(4E73H);  (* RTE return 0 *)
X                       END;
X                    END;
X                 END;
X                 dev:=ADDRESS(currentprocess^.Oport); (* change to port *)
X                 SETREG(8,dev);
X                 CODE(3f48H,64); (* move.w a0,64(a7) set value in stack *)
X              END;
X           END;
X        END;
X        SETREG(8,tbiosSave);  (* move trap adr *)
X        CODE(43faH,10); (* lea 12(pc),a1 *)
X        CODE(2288H); (* move.l a0,(a1) *)
X        CODE(4cdfH,7ffeH); (* restore regs movem *)
X        CODE(4ef9H,0,0) (* jmp back to routine *)
XEND             changedevbios;
X(*$P+,$S+ *)
X
X(*$P-,$S- *)
XPROCEDURE       setup;
XBEGIN
X        tbios:=ADDRESS(changedevbios);
X        CODE(4e75H);    (* RTS *)
XEND             setup;
X(*$P+,$S+ *)
X
X(*$P-,$S- *)
XPROCEDURE       tbiossetup;
XBEGIN
X        tbiosSave:=tbios;
X        CODE(4e75H);    (* RTS *)
XEND             tbiossetup;
X(*$P+,$S+ *)
X(* end device routines *)
X
XPROCEDURE       SelectPort;
XBEGIN
X        currentprocess^.Oport:=Oportdevice;
X        currentprocess^.Iport:=Iportdevice;
X        IF OpenCLI>0 THEN SuperExec(setup) END;
XEND             SelectPort;
X
XPROCEDURE       RunProgram;
XVAR     temp    :       String;
X        p       :       CARDINAL;
XBEGIN
X        IF (bprunning>1) THEN
X           gemsaveGvec^:=ADR(currentprocess^.bpsave);
X           currentprocess^.bpsave:=bpsave;
X           currentprocess^.bpsave[0]:=ADR(currentprocess^.bpsave);
X        END;
X        SetDrvPath(requestdrv,requestpath);
X        IF Pos(currentprocess^.ipname,".",0,p) THEN
```

```
X                ExecProgram;
X        ELSE
X                temp:=currentprocess^.ipname;
X                Concat(currentprocess^.ipname,".prg",
X                       currentprocess^.ipname);
X                ExecProgram;
X                IF currentprocess^.return=(-33) THEN
X                        currentprocess^.ipname:=temp;
X                        Concat(currentprocess^.ipname,".tos",
X                               currentprocess^.ipname);
X                        ExecProgram;
X                END;
X                IF currentprocess^.return=(-33) THEN
X                        currentprocess^.ipname:=temp;
X                        Concat(currentprocess^.ipname,".ttp",
X                        currentprocess^.ipname);
X                        ExecProgram;
X                END;
X        END;
XEND             RunProgram;
X
X(*$P- *)
XPROCEDURE       ExecProgram;
XBEGIN
X          CODE(48e7H,0fffeH); (* save regs *)
X          currentprocess^.tmpcor:=PROCESS(REGISTER(15));
X          SETREG(8,ADR(currentprocess^.ipenvstr));
X          CODE(2f08H);  (* move.l a0,-(sp) *)
X          SETREG(8,ADR(currentprocess^.iptail));
X          CODE(2f08H);  (* move.l a0,-(sp) *)
X          SETREG(8,ADR(currentprocess^.ipname));
X          CODE(2f08H);  (* move.l a0,-(sp) *)
X          CODE(3f3cH,0);    (* move.w #0,-(sp) LOADEXECUTE *)
X          CODE(3f3cH,4bH);  (* move.w #4b,-(sp) gemdos EXEC *)
X          CODE(4e41H);      (* trap #1 *)
X          SETREG(8,currentprocess^.tmpcor);
X          CODE(2e48H);   (* move.l a0,a7 *)
X          CODE(4cdfH,7fffH); (* restore regs *)
X          currentprocess^.return:=INTEGER(REGISTER(0));
X          CODE(4e75H);    (* rts *)
XEND             ExecProgram;
X(*$P+ *)
X
XPROCEDURE       IP;
XBEGIN
X        currentprocess^.ipname:=name;
X        currentprocess^.iptail:=tail;
X        currentprocess^.ipenvstr:=envstr;
X        SelectPort;
X        INC(OpenCLI);
X        INC(bprunning);
X        LOOP
X                MultiBegin;
```

```
X               RunProgram;
X       END;
XEND    IP;
X
XPROCEDURE      BP;
XBEGIN
X       currentprocess^.ipname:=name;
X       currentprocess^.iptail:=tail;
X       currentprocess^.ipenvstr:=envstr;
X       INC(bprunning);
X       SelectPort;
X       MultiBegin;
X
X       RunProgram;
X       MultiEnd;
X       DEC(bprunning);
X       IF currentprocess^.return#0 THEN
X          currentprocess^.errno:=2;
X       ELSE
X          currentprocess^.errno:=0;
X       END;
X       TermProcess(currentprocess^.pid);
XEND    BP;
X
XPROCEDURE      Use;
XVAR            i       :          CARDINAL;
X               pid     :          INTEGER;
X               s0      :          SIGNAL;
X
XPROCEDURE      CheckPort(VAR portdevice: devicetype);
XBEGIN
X          nxparm;
X          ltext(ADR(temp),SIZE(temp));
X          IF temp[0]='-' THEN
X               portdevice:=con;
X               IF temp[1]='n' THEN
X                       portdevice:=null;
X               END;
X               IF temp[1]='a' THEN
X                       portdevice:=aux;
X               END;
X               IF temp[1]='m' THEN
X                       portdevice:=midi;
X               END;
X               IF temp[1]='p' THEN
X                       portdevice:=printer;
X               END;
X               IF temp[1]='0' THEN
X                       portdevice:=dev0;
X               END;
X               IF temp[1]='1' THEN
X                       portdevice:=dev1;
X               END;
```

```
X               IF temp[1]='2' THEN
X                       portdevice:=dev2;
X               END;
X               IF temp[1]='3' THEN
X                       portdevice:=dev3;
X               END;
X          ELSE
X               portdevice:=con;
X               bkparm;
X          END;
XEND            CheckPort;
X
XPROCEDURE      Reset;
XBEGIN
X          request.req:=FALSE;
X          inuse:=FALSE;
XEND            Reset;
X
XPROCEDURE      gettail;
XBEGIN
X        nxparm;
X        etext(ADR(tail[1]),SIZE(tail));
X        bkparm;
X        etext(ADR(envstr),SIZE(envstr));
X        tail[0]:=CHAR(Length(envstr));
X        envstr:='';
XEND            gettail;
X
XPROCEDURE      Caps(VAR str: String); (* convert str to CAPS *)
XVAR               i               :       INTEGER;
XBEGIN
X        i:=0;
X        WHILE ORD(str[i])#0 DO
X              str[i]:=CAP(str[i]);
X              INC(i);
X        END;
XEND            Caps;
X
XBEGIN
X        IF request.magic#MAGIC THEN
X           request.magic:=0;
X           currentprocess^.errno:=54;
X           Reset;
X           RETURN;
X        END;
X        request.magic:=0;
X        cmdstring:=command;
X        inuse:=TRUE;
X        scinit(ADR(cmdstring),SIZE(cmdstring));
X        nxparm;
X        ltext(ADR(name),SIZE(name));
X        Caps(name);
X        IF Compare("IP",name)=Equal THEN
```

```
X           nxparm;
X           ltext(ADR(name),SIZE(name));
X
X           gettail;
X
X           pri:=5;
X           FOR i:=0 TO 79 DO
X               pname[i]:=name[i];
X           END;
X           Reset;
X           Oportdevice:=con;
X           Iportdevice:=con;
X           proc:=IP;
X           sizewsp:=1000;
X           SuperExec(getvector);
X           StartProcess(proc,sizewsp,pri,pname,par);
X           RETURN;
X        END;
X        IF Compare("BP",name)=Equal THEN
X           CheckPort(Iportdevice);
X           CheckPort(Oportdevice);
X           nxparm;
X           ltext(ADR(temp),SIZE(temp));
X           ConvertToInteger(temp,done,pri);
X           IF NOT done THEN
X              pri:=4;
X              bkparm;
X           END;
X           nxparm;
X           ltext(ADR(name),SIZE(name));
X
X           gettail;
X
X           FOR i:=0 TO 79 DO
X               pname[i]:=name[i];
X           END;
X           Reset;
X           proc:=BP;
X           sizewsp:=2000;
X           SuperExec(getbiosvector);
X           StartProcess(proc,sizewsp,pri,pname,par);
X           RETURN;
X        END;
X        IF Compare("FP",name)=Equal THEN
X           CheckPort(Iportdevice);
X           CheckPort(Oportdevice);
X           nxparm;
X           ltext(ADR(temp),SIZE(temp));
X           ConvertToInteger(temp,done,pri);
X           IF NOT done THEN
X              pri:=5;
X              bkparm;
X           END;
```

```
X           nxparm;
X           ltext(ADR(name),SIZE(name));
X
X           gettail;
X
X           FOR i:=0 TO 79 DO
X               pname[i]:=name[i];
X           END;
X           Reset;
X           proc:=BP;
X           sizewsp:=2000;
X           SuperExec(getvector);
X           StartProcess(proc,sizewsp,pri,pname,par);
X           RETURN;
X        END;
X        IF Compare("PORT",name)=Equal THEN
X           nxparm;
X           ltext(ADR(name),SIZE(name));
X           ConvertToInteger(name,done,pid);
X           IF (NOT done) OR (NOT (pid > clipid)) THEN
X              Reset;
X              RETURN;
X           END;
X           s0:=currentprocess;
X           LOOP
X              s0:=s0^.next;
X              IF s0^.pid=1 THEN Reset; RETURN; END;
X              IF s0^.pid=pid THEN EXIT END;
X           END;
X           CheckPort(s0^.Iport);
X           CheckPort(s0^.Oport);
X           Reset;
X           RETURN;
X        END;
X        IF Compare("CRON",name)=Equal THEN
X           nxparm;
X           ltext(ADR(name),SIZE(name));
X           Caps(name);
X           IF Compare("ON",name)=Equal THEN
X              CronActive:=TRUE;
X              LoadCRON;
X           END;
X           IF Compare("OFF",name)=Equal THEN
X              CronActive:=FALSE;
X           END;
X           Reset;
X           RETURN;
X        END;
X        IF Compare("NICE",name)=Equal THEN
X           nxparm;
X           ltext(ADR(name),SIZE(name));
X           ConvertToInteger(name,done,pri);
X           IF NOT done THEN
```

```
X              Reset;
X              RETURN;
X           END;
X           nxparm;
X           ltext(ADR(name),SIZE(name));
X           IF state.return=0 THEN
X              ConvertToInteger(name,done,pid);
X              IF NOT done THEN
X                 pid:=request.pid;
X              END;
X           ELSE
X              pid:=request.pid;
X           END;
X           ChangeProcessPriority(pid,pri);
X           Reset;
X           RETURN;
X        END;
X        IF Compare("HP",name)=Equal THEN
X           nxparm;
X           ltext(ADR(name),SIZE(name));
X           ConvertToInteger(name,done,pid);
X           IF done THEN
X              SleepProcess(pid);
X           ELSE
X              currentprocess^.errno:=3;
X           END;
X           Reset;
X           RETURN;
X        END;
X        IF Compare("WP",name)=Equal THEN
X           nxparm;
X           ltext(ADR(name),SIZE(name));
X           ConvertToInteger(name,done,pid);
X           IF done THEN
X              WakeupProcess(pid);
X           ELSE
X              currentprocess^.errno:=3;
X           END;
X           Reset;
X           RETURN;
X        END;
X        IF Compare("KILL",name)=Equal THEN
X           nxparm;
X           ltext(ADR(name),SIZE(name));
X           ConvertToInteger(name,done,pid);
X           IF done AND (pid>clipid) THEN
X              TermProcess(pid);
X           ELSE
X              currentprocess^.errno:=3;
X           END;
X           Reset;
X           RETURN;
X        END;
```

```
X     Reset;
XEND     Use;
X
X(* return time in seconds *)
XPROCEDURE       converttime(time: CARDINAL): LONGCARD;
XVAR                h,m,s           :         LONGCARD;
XBEGIN
X        h:=LONGCARD(WShr(WAnd(time,63488),11));  (* hours *)
SHAR_EOF
echo "End of part 3, continue with part 4"
echo "4" > s2_seq_.tmp
exit 0


From -9155773891860644885
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,cb2d705c19b41191,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-12-27 00:08:08 PST
Path: utzoo!attcan!uunet!husc6!bloom-beacon!tut.cis.ohio-state.edu!rutgers!ucsd!
ucbvax!agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i084: mx2v230 -- Multitasking kernel with utilities  part04/08
Keywords: shar, Modula-2
Message-ID: <5868@saturn.ucsc.edu>
Date: 27 Dec 88 08:08:08 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1016
Approved: koreth@ssyx.ucsc.edu

Submitted-by: madsen@sask.usask.ca (Jorgen Madsen)
Posting-number: Volume 1, Issue 84
Archive-name: mx2v230/part04

#!/bin/sh
# this is part 4 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file MX2.MOD continued
#
CurArch=4
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> MX2.MOD
X        m:=LONGCARD(WShr(WAnd(time,2016),5));    (* minutes *)
X        s:=2*LONGCARD(WAnd(time,31));            (* seconds *)
```

```
X        RETURN s+(m*60)+(h*3600);
XEND             converttime;
X
X(*$P- *)
XPROCEDURE       gettick;
XBEGIN
X        temphz200:=hz200;
X        CODE(4e75H); (* rts *)
XEND     gettick;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE       setvector;
XBEGIN
X        linea:=s0^.gemsave[1];
X        gemdos:=s0^.gemsave[2];
X        gsxgem:=s0^.gemsave[3];
X        tbios:=s0^.gemsave[4];
X        xbios:=s0^.gemsave[5];
X        linef:=s0^.gemsave[6];
X        level2:=s0^.gemsave[7];
X        level4:=s0^.gemsave[8];
X        shellp:=s0^.gemsave[9];
X        currentprocess^.Oport:=s0^.Oport;
X        currentprocess^.Iport:=s0^.Iport;
X        CODE(4e75H); (* rts *)
XEND     setvector;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE       getvector;
XBEGIN
X        SetDrvPath(defaultdrv,defaultpath);
X        linea:=gemsave[1];
X        gemdos:=gemsave[2];
X        gsxgem:=gemsave[3];
X        tbios:=gemsave[4];
X        xbios:=gemsave[5];
X        linef:=gemsave[6];
X        level2:=gemsave[7];
X        level4:=gemsave[8];
X        shellp:=gemsave[9];
X        currentprocess^.Oport:=devicetype(gemsave[10]);
X        currentprocess^.Iport:=devicetype(gemsave[10]);
X        CODE(4e75H); (* rts *)
XEND     getvector;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE       getbiosvector;
XBEGIN
X        tbios:=gemsave[4];
X        currentprocess^.Oport:=devicetype(gemsave[10]);
```

```
X        currentprocess^.Iport:=devicetype(gemsave[10]);
X        CODE(4e75H); (* rts *)
XEND     getbiosvector;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE       savevector;
XBEGIN
X        GetPath(defaultpath,0);
X        GetDrv(defaultdrv);
X        requestdrv:=defaultdrv;
X        requestpath:=defaultpath;
X        gemsave[1]:=linea;
X        gemsave[2]:=gemdos;
X        gemsave[3]:=gsxgem;
X        gemsave[4]:=tbios;
X        gemsave[5]:=xbios;
X        gemsave[6]:=linef;
X        gemsave[7]:=level2;
X        gemsave[8]:=level4;
X        gemsave[9]:=shellp;
X        gemsave[10]:=ADDRESS(con);
X        returnadr:=termvec;
X        oldikbd:=ADDRESS(ikbdvec);
X        ikbdvec:=keytrapstart;
X        CODE(4e75H); (* rts *)
XEND     savevector;
X(*$P+ *)
X
XPROCEDURE       readcrontab;
XVAR             result,i                          : INTEGER;
X                S                                 : Stream;
X                entry,parm                        : String;
XBEGIN
X        OpenStream(S,"CRONTAB",READ,result);
X        IF result=0 THEN
X           WHILE NOT EOS(S) DO
X                 REadString(S,entry);
X                 scinit(ADR(entry),SIZE(entry));
X                 nxparm;
X                 ltext(ADR(parm),SIZE(parm));
X                 ConvertFromString(parm,10,FALSE,MAX(LONGCARD),
X                                   crontable[i].stime,done);
X                 crontable[i].stime:=crontable[i].stime*60;
X                 nxparm;
X                 ltext(ADR(parm),SIZE(parm));
X                 ConvertFromString(parm,10,FALSE,MAX(LONGCARD),
X                                   crontable[i].freq,done);
X                 crontable[i].freq:=crontable[i].freq*60;
X                 nxparm;
X                 etext(ADR(parm),SIZE(parm));
X                 crontable[i].command:=parm;
X                 crontable[i].active:=TRUE;
```

```
X                INC(i);
X           END;
X           CloseStream(S,result);
X        ELSE
X           CronActive:=FALSE;
X        END;
XEND     readcrontab;
X
XPROCEDURE       LoadCRON; (* crontable loader *)
XBEGIN
X  SuperExec(gettick);
X  cronslice:=temphz200;
X  GetTime(i);
X  currenttime:=converttime(i); (* convert time to seconds *)
X  ticktime:=LONGINT(temphz200 DIV 200)-LONGINT(currenttime);
X       (* ticktime is 200hz clock at 00:00    *)
X  FOR i:=0 TO 15 DO               (* clear crontable *)
X      crontable[i].active:=FALSE;
X  END;
X  readcrontab;
X  FOR i:=0 TO 15 DO
X    IF crontable[i].active THEN
X     IF currenttime>crontable[i].stime THEN
X        periods:=((currenttime-crontable[i].stime) DIV crontable[i].freq)+1;
X
crontable[i].btime:=LONGCARD(ticktime+LONGINT(periods*crontable[i].freq));
X        crontable[i].btime:=(crontable[i].stime+crontable[i].btime)*200;
X     ELSE
X        crontable[i].btime:=LONGCARD(ticktime+LONGINT(crontable[i].stime));
X        crontable[i].btime:=crontable[i].btime*200;
X     END;
X    END;
X  END;
XEND              LoadCRON;
X
XPROCEDURE       TIMER;
XVAR             i       :       CARDINAL;
XBEGIN
X     IF NOT request.req THEN
X      LOOP;
X        FOR i:=0 TO 15 DO
X            IF crontable[i].active THEN
X               IF currentprocess^.slice>crontable[i].btime THEN
X                  REPEAT (* advance to next time slot *)
X                        INC(crontable[i].btime,(crontable[i].freq*200));
X                  UNTIL crontable[i].btime>currentprocess^.slice;
X                  command:=crontable[i].command;
X                  request.magic:=MAGIC;
X                  request.pid:=currentprocess^.pid;
X                  currentprocess^.ipenvstr:=defaultpath;
X                  currentprocess^.flags[0]:=LONGCARD(defaultdrv);
X                  request.req:=TRUE;
X                  EXIT; (* loop *)
```

```
X               END;
X            END;
X        END;
X        EXIT; (* loop *)
X      END; (* loop *)
X     END; (* if *)
XEND     TIMER;
X
XPROCEDURE       HOTKEYER;
XVAR             i,pid,t         :       INTEGER;
XBEGIN
X    IF swloaded THEN
X       s0:=currentprocess;
X       REPEAT
X             s0:=s0^.next
X       UNTIL s0^.pid=cli1;
X       IF s0^.wsp=NIL THEN
X          BConOut(CON,33C);
X          BConOut(CON,'f');
X          screensave^[1]:=physcreen^;
X          physcreen^:=screensave^[2];
X          screensave^[2]:=screensave^[1];
X          BConOut(CON,33C);
X          BConOut(CON,'e');
X          WakeupProcess(clipid);
X          swloaded:=FALSE;
X          done:=Free(ADDRESS(screensave));
X          RETURN;
X       END;
X       BConOut(CON,33C);
X       BConOut(CON,'f');
X       screensave^[1]:=physcreen^;
X       physcreen^:=screensave^[2];
X       screensave^[2]:=screensave^[1];
X       BConOut(CON,33C);
X       BConOut(CON,'e');
X       SleepProcess(cli1);
X       WakeupProcess(cli2);
X       t:=cli1;
X       cli1:=cli2;
X       cli2:=t;
X    END;
X    IF (NOT swloaded) THEN
X       physcreen:=ScreenPhysicalBase();
X       Alloc(64000,screensave);
X       IF ADDRESS(screensave)=NIL THEN RETURN END;
X       BConOut(CON,33C);
X       BConOut(CON,'f');
X       screensave^[2]:=physcreen^;
X       SleepProcess(clipid);
X       cli1:=NextPid(); (* get the pid for the new cli *)
X       cli2:=clipid;
X       BConOut(CON,33C);
```

```
X       BConOut(CON,'e');
X       WriteLn;
X       WriteString("Enter name of program to run: ");
X       WriteLn;
X       WriteString("Press RETURN to run CLI ");
X       ReadString(command);
X       IF command[0]=0c THEN
X          command:="cli";
X       END;
X       Concat("fp ",command,command);
X       tail:="";
X       envstr:="";
X       request.magic:=MAGIC;
X       request.req:=TRUE;
X       Use;
X       swloaded:=TRUE;
X    END;
XEND             HOTKEYER;
X
XPROCEDURE       SysGen;
XBEGIN
X  initprg:="IP CLI.PRG";
X  SYSMEM:=7D00H; (* Allocated memory for MX2 use *)
X  HotKey:=320000H; (* ALT m *)
X  NorMouse:=310000H; (* ALT n *)
X  CurMouse:=2E0000H; (* ALT c *)
X  RebootKey:=130000H; (* ALT r *)
X  memreserve:=7D00H; (* reserved memory for alt HOTKEY program *)
X  ReadMX2INF;
XEND             SysGen;
X
XPROCEDURE       ReadMX2INF;
XVAR             result                          : INTEGER;
X                S                               : Stream;
X
XPROCEDURE       getparm(VAR p: LONGCARD); (* read in info file *)
XVAR     V       :       ADDRESS;
XBEGIN
X     REadString(S,temp);
X     ConvertToAddr(temp,done,V);
X     p:=LONGCARD(V);
XEND             getparm;
X
XBEGIN
X        OpenStream(S,"MX2.INF",READ,result);
X        IF result=0 THEN
X           REadString(S,initprg); (* get command *)
X           getparm(SYSMEM);
X           getparm(HotKey);
X           getparm(NorMouse);
X           getparm(CurMouse);
X           getparm(RebootKey);
X           getparm(memreserve);
```

```
X           CloseStream(S,result);
X        END;
XEND     ReadMX2INF;
X
X(*$P-,$S- *)
XPROCEDURE       initsetup;
XBEGIN
X        currentprocess^.Oport:=con;
X        currentprocess^.Iport:=con;
X        CODE(4e75H);    (* RTS *)
XEND             initsetup;
X(*$P+,$S+ *)
X
X(* ---------------------------------------------------------------------- *)
X
XPROCEDURE       init;
XBEGIN
X  WriteString(TDI);
X  WriteLn;
X  Initsked;
X
X  MultiEnd;
X  SuperExec(initsetup);
X  Alloc(memreserve+2,cmd); (* use spare address vars to setup memory block *)
X  Alloc(2,dev);
X  reservemem:=Free(cmd);
X  OpenCLI := 0;
X
X  MultiEnd;
X  spawnpid:=NextPid();
X  command:="BP 1 spawn";
X  request.pid:=currentprocess^.pid;
X  request.magic:=MAGIC;
X  request.req:=TRUE;
X  MultiEnd;
X  Use; (* execute command *)
X
X  MultiEnd;
X  clipid:=NextPid();
X  command:=initprg;
X  request.pid:=currentprocess^.pid;
X  request.magic:=MAGIC;
X  request.req:=TRUE;
X  MultiEnd;
X  Use; (* execute command *)
X  REPEAT
X        SwapProcess;
X  UNTIL OpenCLI>0;
X
X  CRON:=TIMER;
X  CronActive:=TRUE;
X  LoadCRON; (* read CRONFILE and set up crontable variables *)
X  kbdiorec:=IORec(Keyboard);
```

```
X  ibuf:=kbdiorec^.ibuf;
X  SuperExec(tbiossetup);
X
X LOOP                   (* main kernel loop runs "forever" *)
X     MultiEnd;
X     Kshift:=GetKBShift();
X     IF Kshift=CapsL THEN
X        IF (NOT caps) THEN
X           caps:=TRUE;
X           BConOut(CON,33C);
X           BConOut(CON,'j');
X           BConOut(CON,33C);
X           BConOut(CON,'Y');
X           BConOut(CON,CHAR(32));
X           BConOut(CON,CHAR(111));
X           BConOut(CON,'*');
X           BConOut(CON,33C);
X           BConOut(CON,'k');
X        END;
X     ELSE
X        IF caps THEN
X           caps:=FALSE;
X           BConOut(CON,33C);
X           BConOut(CON,'j');
X           BConOut(CON,33C);
X           BConOut(CON,'Y');
X           BConOut(CON,CHAR(32));
X           BConOut(CON,CHAR(111));
X           BConOut(CON,' ');
X           BConOut(CON,33C);
X           BConOut(CON,'k');
X        END;
X     END;
X
X        IF swapcli THEN
X           swapcli:=FALSE;
X           HOTKEYER;
X        END;
X     IF currentprocess^.slice>cronslice+6000 THEN (* every 30 SECONDS *)
X        cronslice:=currentprocess^.slice;
X
X        IF CronActive THEN
X           SuperExec(getvector);
X           CRON
X        END;
X     END;
X     IF CARDINAL(spintenable)#0 THEN (* check for spints *)
X        FOR i:=0 TO 15 DO (* check all spints and run if set *)
X            IF (i IN spintenable)
X               AND (i IN spintmask)
X               AND (ADDRESS(spint[i].proc)#NIL) THEN
X               spint[i].proc;
X            END;
```

```
X            EXCL(spintenable,i); (* clear flag after complete *)
X         END;
X      END;
X
X      IF request.req THEN
X         s0:=currentprocess;
X         REPEAT
X               s0:=s0^.next
X         UNTIL s0^.pid=request.pid;
X         i:=CARDINAL(s0^.flags[0]);
X         SetDrvPath(i,s0^.ipenvstr);
X         requestdrv:=i;
X         requestpath:=s0^.ipenvstr;
X         SuperExec(setvector);
X         Use;
X         request.req:=FALSE;
X      END;
X      MultiBegin;
X      SwapProcess;
X   END;
XEND     init;
X
X(* ------------------------------------------------------------------- *)
X
XBEGIN
X         SuperExec(savevector);
X         BConOut(CON,33C);
X         BConOut(CON,'E');
X         Hotset:=KBShifts{AlternateKey};
X         CapsL:=KBShifts{CapsLock};
X         IF (ROMDATE # OLDDATE) AND (ROMDATE # NEWDATE) THEN
X            WriteLn;
X            WriteString("SORRY, MX2 MAY NOT RUN WITH YOUR ROM VERSION.");
X            WriteLn;
X         END;
X         SysGen; (* read in system generation file if any *)
X         IF CreateHeap(SYSMEM,TRUE) THEN
X            sysmemsize:=SYSMEM;
X            Oportdevice:=con;
X            Iportdevice:=con;
X            inuse:=FALSE;
X            done:=TRUE;
X            InitProcesses;
X            request.pid:=currentprocess^.pid;
X            MultiEnd;
X            WriteLn;
X            WriteString(TITLE1);
X            WriteString(VERSION);
X            WriteString(TITLE2);
X            WriteLn;
X            proc:=init;
X            sizewsp:=2000;
X            pri:=1;
```

```
X           pname:="init";
X           par:=NIL;
X           StartProcess(proc,sizewsp,pri,pname,par);
X        END;
X        OldTerm;
XEND MX2.
X
SHAR_EOF
chmod 0600 MX2.MOD || echo "restore of MX2.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > MX2LIB.C &&
X/* These are some interface routines for MX2 using C. It is written     */
X/* using Mark Johnson's Shareware C compiler version 1.2                */
X/* If I receive requests maybe I will write a complete interface or     */
X/* someone else can feel free.                                          */
X
X
X#define MX2GLOBAL       0x144L
X#define MX2MAGIC        22261L
Xstruct  PIPE    {
X        char    pipename[80];
X        short   buf[1024];
X        int     bufsize;
X        int     bufhead;
X        int     buftail;
X        int     cnt;
X};
Xstruct  sysreq  {
X        int     req;
X        int     pid;
X        long    magic;
X};
Xstruct  spinttype {
X        long    proc;
X        int     pid;
X        long    data;
X};
Xstruct  sysvar  {
X        long    *currentprocess;
X        int     *MULTI;
X        long    *slicebegin;
X        long    *contextswitch;
X        long    *command;
X        long    *request;
X        long    *CRON;
X        int     *spintenable;
X        int     *spintmask;
X        long    *spint;
X        long    *bpsave;
X        long    *pipes;
X        long    *sysmemsize;
X        long    *gemsaveGvec;
X        long    *StartProcess; /* The following are pointers to fuctions */
X        long    *SwapProcess;
```

```
X        long    *TermProcess;
X        long    *NextPid;
X        long    *SleepProcess;
X        long    *WakeupProcess;
X        long    *ChangeProcessPriority;
X        long    *MultiBegin;
X        long    *MultiEnd;
X        long    *DozeProcess;
X        long    *WaitProcess;
X};
X        struct  sysvar  *mx2var;
X
Xmain()
X{
X        long    *contextswitch,*sysmemsize;
X
X        GetSysVar(); /* store mx2 system variable pointers in mx2var */
X        contextswitch = (long *) mx2var->contextswitch;
X        printf("Total context switches  %lu   / address %lx / \n",
X                *contextswitch,contextswitch);
X        sysmemsize = (long *) mx2var->sysmemsize;
X        printf("MX2 system memory  %lu   / address %lx / \n\n",
X                *sysmemsize,sysmemsize);
X        exit(0);
X}
X
X/*      This routine will read MX2 system variables and procedures      */
X/*      and return them as a pointer in mx2var.                         */
Xvoid    GetSysVar()
X{
X        long    *adr,ssv;
X
X        ssv = 0L;
X        ssv = (long *) trap(1,0x20,ssv);
X        adr = MX2GLOBAL;           /* system variable address pointer */
X        mx2var = (struct sysvar *) *adr;
X        ssv = trap(1,0x20,ssv);
X}
SHAR_EOF
chmod 0600 MX2LIB.C || echo "restore of MX2LIB.C fails"
sed 's/^X//' << 'SHAR_EOF' > MX2NET.MOD &&
X
X(*              Copyright 1987 fred brooks LogicTek             *)
X(*                                                              *)
X(*                                                              *)
X(*   First Release                      12/8/87-FGB             *)
X(*                                                              *)
X
XMODULE mx2net; (*$S-,$T-,$A+ *)
XFROM    SYSTEM  IMPORT  CODE,ADDRESS;
XFROM    NETWORK IMPORT  initnetwork,recframe;
XFROM    GEMX    IMPORT  BasePageAddress;
XFROM    GEMDOS  IMPORT  TermRes,Super;
```

```
XFROM    BIOS    IMPORT  Device;
XVAR
X       vblptr [456H]                           : POINTER TO ARRAY [0..7]
X                                                 OF ADDRESS;
X       i                                       : CARDINAL;
X       ssv                                     : ADDRESS;
X
X(*      it runs as a background process in a vbl time slot              *)
X(*$P- *)
XPROCEDURE VBLrecframe;
XBEGIN
X        CODE(02f39H,0,04a2H); (* move.l $4a2,-(sp) save BIOS pointer *)
X        CODE(04b9H,0,02eH,0,04a2H); (* sub 46 from pointer *)
X        recframe; (* check network for data *)
X        CODE(023dfH,0,04a2H); (* restore BIOS pointer *)
X        CODE(4e75H); (* rts *)
XEND     VBLrecframe;
X(*$P+ *)
X
XBEGIN
X        initnetwork(HSS);
X        i:=0;
X        ssv:=0;
X        Super(ssv);
X        WHILE vblptr^[i]#0H DO
X              INC(i);
X        END;
X         (* set up vbl vector for NETWORK input *)
X        vblptr^[i] := ADDRESS(VBLrecframe);
X        Super(ssv);
X    WITH BasePageAddress^ DO
X    TermRes(CodeLen+BssLen+LONGCARD(CodeBase-ADDRESS(BasePageAddress)),0);
X    END;
XEND     mx2net.
X
SHAR_EOF
chmod 0600 MX2NET.MOD || echo "restore of MX2NET.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > MX2NOTES.TXT &&
Xnotes.txt
XThese are working notes to help keep track of changes or things that
Xshould or could be done to MX2.
X
X
X#@MX2 file and program directorys
XThere are two new directory that MX2 needs. The first is called \MX2SPOOL
Xthis is created when the spooler program begins for the first time.
XThe second is called \MX2 these should be where all MX2 executable
Xprograms should be. All script file should now look at \MX2 for MX2
Xprograms that they need.
X
X#@mlimit
XNeed a way to limit the amount that any one program can see with a
Xmalloc(-1). I could set up a trap handler to control the malloc but I will
```

```
Xfirst try to allocate small 2 byte memory blocks to break up the memory
Xblock.
X
X#@crashs
XWhen MX2 crashes disk sectors that were being allocated for a file that did
Xnot close can be lost. Use of a program like "TUNEUP" can fix and get back
Xthe "LOST" sectors. Also some auto programs can cause problems with MX2
XI use GEMBOOT.PRG and the mitchtron CACHE.PRG. I have found that the
XL-CACHE.PRG causes lockup and random aborts.
X
X#@xmodem
XNeed to take a look to see why the xmodem routines are messing up later
XI/O with the te command and CLI redirection to the AUX port.
X
X#@menu
XI have started to play with a backgound gem menu bar for the CLI. The
Xprogram MENU sets up a menu bar that can be run in backgound with the
X"sys bp 1 menu" command. The program 'km' will terminate the MENU
Xprogram. Although it works it also totally screws up gem. It's safer run
Xas just a regular program but it you don't care about running another gem
Xprogram try it.
X
X#@WaitProcess MX2V220
XProcedure that waits for a memory location to change it's value.
X   WaitProcess(VAR id: INTEGER; VAR location: ADDRESS; VAR value,mask,msec:
LONGCARD);
XThis procedure puts the currentprocess to sleep then compares the contents
Xat 'location' against 'value AND mask'. If they are the same then the
Xprocess wakes up and is put in the scheduler ready list to be run. msec
Xis the time the process will wait for the value in milliseconds if msec
Xis set to 0 WaitProcess will never timeout.
X** NEED TO ADD TO JSM2 syscall.def
X
X#@submita&submitm
XFixed submita and added submitm to utility programs. I forgot to use the
Xxbios call to get the IOREC for the RS232 buffer.
X
X@#mx2.acc
XIf you want to switch out of a GEM program back to the ALT cli with the
XALT m hotkey you MUST turn off the MX2.ACC first. If you don't the AES
Xwill get very mixed up. When back in the GEM program you can turn the
XMX2.ACC back on to allow normal background program operation. This guy is
Xcauseing some problems I may have to redesign it again.
X
X@#spint control of spooler, network and xmodem
Xspint 0 controls the spooler
Xspint 1 controls both netaux and netmidi
Xspint 2 controls the xmodem send and receive
Xxr = xmodem receive  'xr -crc foo.bar' or 'xr -chk foo.bar'
Xxt = xmodem send     'xt -crc foo.bar' or 'xr -chk foo.bar'
Xxs = xmodem status   'xs'
Xxa = xmodem abort    'xa'
X
```

```
X@#spints
XSpints in MX2 are similar to signals in UNIX. They are procedures that
Xcan be started by issueing a Trigger(x) call. A spint can also carry a
Xmemory location that can be used for shared memory between processes.
XThe source files spooler.mod, sq.mod, lp.mod and killlp.mod are simple
Xexamples.
Xspooler.mod uses the EnableSpint call to setup spint 0 to a procedure
Xsq and a pointer to the variable spintcmd.
Xsq.mod, lp.mod and killlp.mod use the SpintInfo and Trigger calls to
Xcommunicate with the spooler.prg
X
X
SHAR_EOF
chmod 0600 MX2NOTES.TXT || echo "restore of MX2NOTES.TXT fails"
sed 's/^X//' << 'SHAR_EOF' > NETAUX.MOD &&
X(*$T-,$S-,$A+ *)
XMODULE  netaux;
XFROM    NETWORK         IMPORT  initnetwork,recframe;
XFROM    BIOS            IMPORT  Device;
XFROM    SYSCALL         IMPORT  SwapProcess;
XBEGIN
X        initnetwork(AUX);
X        LOOP
X                recframe;
X                SwapProcess;
X        END;
XEND     netaux.
X
SHAR_EOF
chmod 0600 NETAUX.MOD || echo "restore of NETAUX.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > NETMIDI.MOD &&
X(*$T-,$S-,$A+ *)
XMODULE  netmidi;
XFROM    NETWORK         IMPORT  initnetwork,recframe;
XFROM    BIOS            IMPORT  Device;
XFROM    SYSCALL         IMPORT  SwapProcess;
XBEGIN
X        initnetwork(HSS);
X        LOOP
X                recframe;
X                SwapProcess;
X        END;
XEND     netmidi.
X
SHAR_EOF
chmod 0600 NETMIDI.MOD || echo "restore of NETMIDI.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > NETWORK.DEF &&
X
X(*              Copyright 1987 fred brooks LogicTek             *)
X(*                                                              *)
X(*                                                              *)
X(*   First Release                      12/8/87-FGB             *)
X(* Added support for use of MIDI or AUX port for network        *)
```

```
X(*                                     2/6/88-FGB              *)
X(*                                                             *)
X
XDEFINITION MODULE NETWORK;
XFROM        BIOS    IMPORT    Device;
XVAR         netdevice         :    Device;
XPROCEDURE       initnetwork(port: Device);
X(* install network, default is network on ,MIDI port *)
XPROCEDURE       recframe;    (* check network for received data *)
XPROCEDURE       networkoff;  (* turn on network *)
XPROCEDURE       networkon;   (* turn off network *)
X
XEND  NETWORK.
SHAR_EOF
chmod 0600 NETWORK.DEF || echo "restore of NETWORK.DEF fails"
sed 's/^X//' << 'SHAR_EOF' > NETWORK.MOD &&
X
X(*              Copyright 1987 fred brooks LogicTek            *)
X(*                                                             *)
X(*                                                             *)
X(*   First Release                      12/8/87-FGB             *)
X(*                                                             *)
X
XIMPLEMENTATION MODULE NETWORK ;
X
X(* ---------------------------------------------------------------------------
X
X               NETWORK : MIDI PORT TWO CPU NETWORK FOR TDI Modula-2/ST
X
X   ---------------------------------------------------------------------------*)
X
X(*$T-,$S-,$A+ *)
X
XFROM SYSTEM IMPORT ADDRESS, ADR, SETREG, CODE, REGISTER ,BYTE ,TSIZE;
XFROM BIOS   IMPORT BPB ,BConStat ,BConIn, BCosStat, BConOut, Device,
X                   MediaChange,MCState,GetBPB,RWAbs,RW,DriveSet,DriveMap;
XFROM XBIOS  IMPORT SuperExec,IORec,IORECPTR,IOREC,SerialDevice;
XFROM GEMDOS IMPORT TermRes,Open,Close ;
XIMPORT             GEMDOS;
XFROM ASCII  IMPORT SYN,STX,SOH,BEL;
X
XCONST
X  MaxSeq          = 1;
X  recsize         = 511;
X  USER            = 324159265;
X  retry           = 10;
X  debug           = FALSE;
X  trace           = FALSE;
X
X  (* Because we dont know what registers the BIOS is using we must use
X     the following opcodes to save the registers *)
X  MOVEMDEC = 48E7H ;    (* 68000 opcode for MOVEM <regs>,-(A7) *)
X  MOVEMINC = 4CDFH ;    (* 68000 opcode for MOVEM (A7)+,<regs> *)
```

```
X  SAVEREGS = 07FFCH ;   (* Registers D1..A5 for DEC *)
X  RESTREGS = 03FFEH ;   (* Registers D1..A5 for INC *)
X  RTS = 04E75H ;        (* 68000 return from subroutine opcode *)
X
XTYPE
X  (* Procedure types to mimic correct sequence for "C" BIOS routines *)
X
X  CBPBProc     = PROCEDURE ( CARDINAL ) ;
X  CMediaChProc = PROCEDURE ( CARDINAL ) ;
X  CRWAbsProc   = PROCEDURE ( CARDINAL, CARDINAL, CARDINAL, ADDRESS, CARDINAL );
X  MIDIbuffer   = ARRAY [0..512] OF CARDINAL;
X  SequenceNr   = [0..MaxSeq];
X  message      = ARRAY [0..recsize] OF BYTE;
X  message1     = ARRAY [0..17] OF BYTE;
X  FrameKind    = (ack,data,callreq,callaccp,clearreq,clearconf,
X                 resetreq,resetconf,diag);
X  DataKind     = (rdmediareq,rdmediaconf,rdbpbreq,rdbpbconf,
X                 rdrwabsreq,rdrwabsconf);
X  evtype       = (framearrival,cksumerr,timeout,hostready,reset,nothing);
X
X  frame        = RECORD
X                  syn    :       CHAR; (* these are sync chars *)
X                  stx    :       CHAR; (* for the frames       *)
X                  kind   :       FrameKind;
X                  seq    :       SequenceNr;
X                  ack    :       SequenceNr;
X                  cmd    :       DataKind;
X                  rw     :       CARDINAL; (* read or write data *)
X                  recno  :       CARDINAL; (* sector for data*)
X                  d0     :       LONGCARD; (* data return variable *)
X                  info   :       message;
X                  user   :       LONGCARD;
X                  cksum  :       CARDINAL;
X                END;
X
X  framecptr    = POINTER TO framecmd;
X  framecmd     = RECORD
X                  syn    :       CHAR; (* these are sync chars *)
X                  stx    :       CHAR; (* for the frames       *)
X                  kind   :       FrameKind;
X                  seq    :       SequenceNr;
X                  ack    :       SequenceNr;
X                  cmd    :       DataKind;
X                  rw     :       CARDINAL; (* read or write data *)
X                  recno  :       CARDINAL; (* sector for data*)
X                  d0     :       LONGCARD; (* data return variable *)
X                  info   :       message1;
X                  user   :       LONGCARD;
X                  cksum  :       CARDINAL;
X                END;
X
X  control     = RECORD
X                  magic          :       LONGCARD;
```

```
X                reset          :       BOOLEAN;
X                networkactive  :       BOOLEAN;
X                remotedrive    :       CARDINAL;
X                drivemap       :       DriveSet;
X               nextframetosend :      SequenceNr;
X               frameexpected   :      SequenceNr;
X               sendreset       :      BOOLEAN;
X              END;
X
X  consave     = RECORD
X                magic          :       LONGCARD;
X                reset          :       BOOLEAN;
X                networkactive  :       BOOLEAN;
X               END;
X
X  frameptr      =       POINTER TO ARRAY [0..1024] OF BYTE;
X
XVAR
X
X
X  (* BIOS variables : These can only be accessed with the 68000 in supervisor
X     mode. The Modula-2 language allows you to fix the location of variables *)
X
X  HDBPB     [0472H] : ADDRESS ;       (* hard disk get Bios Parameter Block *)
X  HDRWAbs   [0476H] : ADDRESS ;       (* hard disk read/write abs   *)
X  HDMediaCh [047EH] : ADDRESS ;       (* hard disk media change     *)
X  DriveBits [04C2H] : SET OF [0..31]; (* disk drives present map    *)
X  flock     [043EH] : LONGCARD;       (* disk access in progress    *)
X  hz200     [04baH] : LONGCARD;       (* 200hz clock counter        *)
X  clock             : LONGCARD;
X  Dptr              : DriveSet;       (* save original drive map    *)
X  Mptr              : LONGCARD;
X  charcount,j,framesize,cksum,recframesize,sndframesize,
X  SIZEframe,SIZEframecmd                                : CARDINAL;
X
X  networkconnect          :   BOOLEAN; (* DCD = 1 TRUE  *)
X  gotframe                :   BOOLEAN;
X  framebufferfull         :   BOOLEAN;
X  cleartosend             :   BOOLEAN;
X  readytosend             :   BOOLEAN;
X  requesttosend           :   BOOLEAN;
X  framewaiting            :   BOOLEAN;
X  timer,OK,installed      :   BOOLEAN;
X  gotmediach              :   ARRAY [0..5] OF BOOLEAN;
X  gotbpb                  :   ARRAY [0..5] OF BOOLEAN;
X  networkerror            :   BOOLEAN;
X  shortframe              :   BOOLEAN;
X  sendlong                :   BOOLEAN;
X
X  sframe,rframe,SFRAME,RFRAME,
X  nframe1,nframe2                  :   frame;
X  rframeptr,sframeptr,
X  bpbptr,nbpbptr                   :   frameptr;
```

```
X  framecmdptr,framecmdptr1         :   framecptr;
X  event                            :   evtype;
X  C                                :   control;
X  recchar,timestart,timefortimeout,timeouttime :   LONGCARD;
X  timestart1,timefortimeout1,timeouttime1      :   LONGCARD;
X  result,r,i,i1,i2,i3,mediacount,handle        :   INTEGER;
X  D0ptr                                        :   POINTER TO LONGCARD;
X  wsector,drvnr,DriveA,DriveF,devicestart,d,R  :   CARDINAL;
X  rbuffer                                      :   MIDIbuffer;
X  rbptr                                        :   IORECPTR;
X  numBytes,sec,min,hour,time,count             :   LONGCARD ;
X  status                                       :   LONGINT ;
X
X  (* The following are saved copies of the BIOS variables so that the real
X     hard disk routines can be called if a hard disk access is requested. *)
X
X  SaveHDBPB      : CBPBProc ;     (* hard disk get Bios Parameter Block *)
X  SaveHDRWAbs    : CRWAbsProc ;   (* hard disk read/write abs *)
X  SaveHDMediaCh  : CMediaChProc ; (* hard disk media change *)
X
X  (* NETWORK control *)
X
X  NetworkBPB  : ARRAY [0..5] OF BPB ; (* BIOS Parameter block for NETWORK *)
X
XPROCEDURE MoveMemory ( From, To : ADDRESS ; Bytes : LONGCARD ) ;
X(* This routine shows how time critical portions of code can be optimised to
X   run faster. It relys on the code generation rules of the compiler which
X   can be checked by dis-assembling the link file with DecLnk.*)
X
XCONST
X  MOVEB = 12D8H ;       (*      MOVE.B  (A0)+,(A1)+     *)
X  MOVEL = 22D8H ;       (*      MOVE.L  (A0)+,(A1)+     *)
X  A0    = 0+8 ;         (* register A0 *)
X  A1    = 1+8 ;         (* register A1 *)
X
XBEGIN
X  SETREG(A0,From) ;             (* load From pointer into A0 *)
X  SETREG(A1,To) ;               (* load To pointer into A1 *)
X
X  IF ( ODD(From) OR ODD(To) ) THEN      (* must do bytes *)
X    WHILE ( Bytes <> 0 ) DO
X      CODE(MOVEB) ;
X      DEC(Bytes) ;
X    END ;
X  ELSE (* even addresses so can do long moves *)
X    WHILE ( Bytes > 3 ) DO
X      CODE(MOVEL) ;
X      DEC(Bytes,4) ;
X    END ;
X    WHILE ( Bytes <> 0 ) DO
X      CODE(MOVEB) ;             (* clean up remainder *)
X      DEC(Bytes) ;
X    END ;
```

```
X  END ;
XEND MoveMemory ;
X
X
XPROCEDURE inc(VAR k: SequenceNr);   (* increment k circulary *)
XBEGIN
X        IF k<MaxSeq THEN k:=k+1 ELSE k:=0 END;
XEND     inc;
X
X
X(* The following procedures mimic the disk handling routines called by the
X   BIOS. Their procedure declarations have been written to mimic the "C"
X   calling sequence. *)
X
XPROCEDURE RDRWAbs ( device, RecordNum, SectorCount : CARDINAL ;
X                    Buffer : ADDRESS ; Flag : CARDINAL ) ;
X(* NB. It is assumed that GEMDOS wont call this routine with out of range
X   parameters *)
XCONST D0 = 0 ;
XBEGIN
X  CODE(MOVEMDEC,SAVEREGS) ;     (* save registers on stack *)
X  status := 0;
X  IF (device >= DriveA) AND (device <=DriveF) THEN (* is NETWORK channel *)
X    IF ( Flag = 0 ) OR ( Flag = 2 ) (* read *)  THEN
X       FOR wsector:=0 TO (SectorCount-1) DO
X           C.remotedrive:=device-devicestart;
X           nframe1.d0:=LONGCARD(device-devicestart);
X           nframe1.recno:=RecordNum+wsector;
X           nframe1.rw:=Flag; (* read *)
X           resetnewdisk;
X           IF getfromremote(rdrwabsreq,rdrwabsconf,nframe1) THEN
X              MoveMemory(ADR(nframe1.info),Buffer+ADDRESS(wsector)*512,
X                         512);
X              status:=0;
X           ELSE
X              status:=(-11);
X           END; (* if *)
X       END; (* for *)
X    IF networkerror THEN C.sendreset:=TRUE END; (* send network reset to remote
cpu *)
X      SETREG(D0,status) ;
X    ELSIF ( Flag = 1 ) OR ( Flag = 3 ) THEN (* write *)
X       FOR wsector:=0 TO (SectorCount-1) DO
X           C.remotedrive:=device-devicestart;
X           nframe1.d0:=LONGCARD(device-devicestart);
X           nframe1.recno:=RecordNum+wsector;
X           nframe1.rw:=Flag; (* write *)
X           resetnewdisk;
X           MoveMemory(Buffer+ADDRESS(wsector)*512,ADR(nframe1.info),512);
X           IF getfromremote(rdrwabsreq,rdrwabsconf,nframe1) THEN
X              status:=0;
X           ELSE
X              status:=(-10);
```

```
X           END;
X       END; (* for *)
X    IF networkerror THEN C.sendreset:=TRUE END; (* send network reset to remote
cpu *)
X      SETREG(D0,status) ;
X    ELSE
X      SETREG(D0,LONGINT(-3)) ;
X    END ;
X  ELSE (* not NETWORK *)
X    SaveHDRWAbs (device,RecordNum,SectorCount,Buffer,Flag) ;
X  END ;
X  CODE(MOVEMINC,RESTREGS) ;     (* Restore registers from stack *)
XEND RDRWAbs ;
X
XPROCEDURE RDMediaCh ( device : CARDINAL ) ;
XCONST D0 = 0 ;
XBEGIN
X  CODE(MOVEMDEC,SAVEREGS) ;     (* save registers on stack *)
X  IF (device >= DriveA) AND (device <=DriveF) THEN (* is NETWORK channel *)
X    C.remotedrive:=device-devicestart;
X    nframe1.d0:=LONGCARD(device-devicestart);
X    IF newdisk() THEN
X       gotmediach[device-devicestart]:=FALSE;
X       gotbpb[device-devicestart]:=FALSE;
X    END;
X    IF (NOT gotmediach[device-devicestart]) THEN
X     IF getfromremote(rdmediareq,rdmediaconf,nframe1) THEN
X        gotmediach[device-devicestart]:=TRUE;
X        IF nframe1.d0=1 THEN nframe1.d0:=2 END;
X        SETREG(D0,nframe1.d0) ;    (* "C" uses D0 as return location *)
X     ELSE
X        SETREG(D0,Changed);
X     END;
X    ELSE
X       SETREG(D0,NoChange) ;    (* "C" uses D0 as return location *)
X    END;
X  ELSE (* not NETWORK *)
X    SaveHDMediaCh(device) ;
X  END;
X  CODE(MOVEMINC,RESTREGS) ;     (* Restore registers from stack *)
SHAR_EOF
echo "End of part 4, continue with part 5"
echo "5" > s2_seq_.tmp
exit 0


From 648674756787344561
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,c5704e31135567eb,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-12-27 00:08:18 PST
Path: utzoo!attcan!uunet!husc6!bloom-beacon!tut.cis.ohio-state.edu!rutgers!ucsd!
ucbvax!agate!saturn!ssyx.ucsc.edu!koreth
```

From: koreth@ssyx.ucsc.edu.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i085: mx2v230 -- Multitasking kernel with utilities part05/08
Keywords: shar, Modula-2
Message-ID: <5869@saturn.ucsc.edu>
Date: 27 Dec 88 08:08:18 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 964
Approved: koreth@ssyx.ucsc.edu

Submitted-by: madsen@sask.usask.ca (Jorgen Madsen)
Posting-number: Volume 1, Issue 85
Archive-name: mx2v230/part05

```
#!/bin/sh
# this is part 5 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file NETWORK.MOD continued
#
CurArch=5
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> NETWORK.MOD
XEND RDMediaCh ;
X
XPROCEDURE RDBPB ( device : CARDINAL ) ;
XCONST D0 = 0 ;
XBEGIN
X  CODE(MOVEMDEC,SAVEREGS) ;     (* save registers on stack *)
X  IF (device >= DriveA) AND (device <=DriveF) THEN (* is NETWORK channel *)
X    C.remotedrive:=device-devicestart;
X    nframe1.d0:=LONGCARD(device-devicestart);
X    IF newdisk() THEN gotbpb[device-devicestart]:=FALSE; gotmediach[device-
devicestart]:=FALSE END;
X(*       gotbpb[device-devicestart]:=FALSE;   (* test *) *)
X    IF (NOT gotbpb[device-devicestart]) THEN
X     IF getfromremote(rdbpbreq,rdbpbconf,nframe1) THEN
X       gotbpb[device-devicestart]:=TRUE;
X       bpbptr:=ADR(nframe1.info);
X       nbpbptr:=ADR(NetworkBPB[device-devicestart]);
X       FOR i3:=0 TO TSIZE(BPB)-1 DO
X           nbpbptr^[i3]:=bpbptr^[i3];
X       END;
X       resetnewdisk;
X       SETREG(D0,ADR(NetworkBPB[device-devicestart])); (* D0 returns address of
the BPB *)
```

```
X     ELSE
X       SETREG(D0,0);
X     END;
X    ELSE
X       SETREG(D0,ADR(NetworkBPB[device-devicestart])); (* D0 returns address of
the BPB *)
X    END;
X    IF networkerror THEN C.sendreset:=TRUE END; (* send network reset to remote
cpu *)
X  ELSE (* not NETWORK *)
X    SaveHDBPB(device) ;
X  END ;
X  CODE(MOVEMINC,RESTREGS) ;     (* Restore registers from stack *)
XEND RDBPB ;
X
XPROCEDURE resetnewdisk;
XBEGIN
X        SuperExec(gettime);
X        timestart1:=clock;
X        timefortimeout1:=timestart1;
X        IncTime(timefortimeout1,2);
XEND     resetnewdisk;
X
XPROCEDURE newdisk(): BOOLEAN;
XBEGIN
X        SuperExec(gettime);
X        timeouttime1:=clock;
X        SETREG(0,timeouttime1);
X        CODE(0280H,0,0FFFFH);
X        timeouttime1:=LONGCARD(REGISTER(0));
X        IF timeouttime1>timefortimeout1 THEN
X           resetnewdisk;
X           RETURN TRUE;
X        END;
X        RETURN FALSE;
XEND     newdisk;
X
X(* ------------------------------------------------------------------------ *)
X
XPROCEDURE Initialise (port: Device) : BOOLEAN ;
X(* returns TRUE if NETWORK is to be installed *)
XBEGIN
X  CODE(3f3cH,0017H,4e4eH,548fH);           (* gettime *)
X  CODE(2f00H,3f3cH,0016H,4e4eH,5c8fH);     (* settime *)
X  IF NOT installed THEN
X    SuperExec(PROC(setcontrol));  (* set address of global control record *)
X  END;
X  IF port=HSS THEN
X    rbptr:=IORec(MIDI);
X  ELSE
X    rbptr:=IORec(RS232);
X  END;
X  rbptr^.ibuf:=ADR(rbuffer);
```

```
X  rbptr^.ibufsize:=1024;
X  C.magic:=USER;
X  C.remotedrive:=0;
X  framesize:=TSIZE(frame);
X  recframesize:=framesize;
X  sndframesize:=framesize;
X  sframe.user:=USER;
X  R:=0;
X  RETURN TRUE;
XEND Initialise ;
X
X(*$P- *) (* set vector to control record *)
XPROCEDURE setcontrol;
XBEGIN
X        IF Mptr#USER THEN
X           C.drivemap:=DriveMap();
X           Dptr:=C.drivemap;
X        END;
X        C.drivemap:=Dptr;
X        Mptr:=USER;
X        CODE(RTS);
XEND     setcontrol;
X
XPROCEDURE nrecframe;
XBEGIN
X        IF C.networkactive THEN
X           WHILE (BConStat(netdevice)) AND (NOT framebufferfull) DO
X                 recchar := BConIn(netdevice);
X                 IF (CHAR(recchar)=SYN) AND (NOT gotframe) THEN
X                    gotframe:=TRUE; (* got sync char from data *)
X                    charcount:=0;
X                 END;
X                 IF (charcount=1) AND ((CHAR(recchar)#STX) AND
(CHAR(recchar)#SOH)) THEN
X                    gotframe:=FALSE; (* false start try again *)
X                    charcount:=0;
X                 END;
X                 IF (charcount=1) AND (CHAR(recchar)=STX) THEN
X                    recframesize:=SIZEframe;
X                 END;
X                 IF (charcount=1) AND (CHAR(recchar)=SOH) THEN
X                    recframesize:=SIZEframecmd;
X                 END;
X                 IF gotframe THEN                  (* put data in buffer *)
X                    rframeptr^[charcount]:=BYTE(recchar);
X                    INC(charcount);
X                    IF charcount=recframesize THEN (* got full frame *)
X                       gotframe := FALSE;
X        IF trace THEN BConOut(CON,"^") END;
X                       IF recframesize=SIZEframecmd THEN
X                          rframe.user:=framecmdptr^.user;
X                          rframe.cksum:=framecmdptr^.cksum;
X                       END;
```

```
X                       framebufferfull := TRUE;
X                    END;
X                 END;
X           END; (* WHILE *)
X        END;
XEND     nrecframe;
X
X(* The following compiler directive stops the compiler from generating the
X   normal Modula-2 entry/exit code for the next procedure. This is needed as
X   this routine is called in supervisor mode by the BIOS function to install
X   the BIOS vectors. *)
X(*$P- Stop entry/exit code for next procedure *)
XPROCEDURE InstallVectors ;
XBEGIN
X  (* First save the current hard disk vectors *)
X  SaveHDBPB := CBPBProc(HDBPB) ;
X  SaveHDRWAbs := CRWAbsProc(HDRWAbs) ;
X  SaveHDMediaCh := CMediaChProc(HDMediaCh) ;
X  (* Now set the BIOS vectors to our routines *)
X  HDBPB := ADDRESS(RDBPB) ;
X  HDRWAbs := ADDRESS(RDRWAbs) ;
X  HDMediaCh := ADDRESS(RDMediaCh) ;
X  drvnr:=2;
X  WHILE drvnr IN DriveBits DO
X        INC(drvnr);
X  END; (* while *)
X  INC(drvnr);
X  devicestart:=drvnr;
X  DriveA:=drvnr;
X  DriveF:=drvnr+5;
X  INCL(DriveBits,drvnr) ;             (* set new drive A *)
X  INCL(DriveBits,drvnr+1) ;           (* set new drive B *)
X  INCL(DriveBits,drvnr+2) ;           (* set new drive C *)
X  INCL(DriveBits,drvnr+3) ;           (* set new drive D *)
X  INCL(DriveBits,drvnr+4) ;           (* set new drive E *)
X  INCL(DriveBits,drvnr+5) ;           (* set new drive F *)
X  networkconnect := FALSE;
X  gotframe := FALSE;
X  framebufferfull := FALSE;
X  charcount:=0;
X  SIZEframe:=TSIZE(frame);
X  SIZEframecmd:=TSIZE(framecmd);
X
X  rframeptr := ADR(rframe);
X  framecmdptr:=ADR(rframe);
X  sframeptr := ADR(sframe);
X  CODE(RTS) ;                (* code to return to calling BIOS function *)
XEND InstallVectors ;
X
XPROCEDURE waitcts(what: BOOLEAN); (* wait for cleartosend state *)
XBEGIN
X        IF what THEN
X           REPEAT
```

```
X           nrecframe;
X           Nwait(event);
X           HandleEvents();
X           IF R>retry THEN
X              networkerror:=TRUE;
X              RETURN; (* trouble *)
X           END;
X           UNTIL cleartosend;
X           RETURN;
X        ELSE
X           LOOP
X           nrecframe;
X           Nwait(event);
X           IF (NOT cleartosend) THEN EXIT END;
X           HandleEvents();
X           IF R>retry THEN
X              networkerror:=TRUE;
X              RETURN; (* trouble *)
X           END;
X           END; (* loop *)
X        IF trace THEN BConOut(CON,"N") END;
X           HandleEvents();
X        END;
XEND     waitcts;
X
X(* request for data from remote hosts disk drives and system *)
X(* what wanted in command, the correct reply in reply, data in f *)
XPROCEDURE getfromremote(command, reply: DataKind; VAR f: frame): BOOLEAN;
XBEGIN
X        IF (NOT C.networkactive) THEN RETURN FALSE END; (* error *)
X        networkerror:=FALSE;
X        R:=0;
X        StartTimer;
X        IF trace THEN BConOut(CON,"A") END;
X        f.kind:=data;
X        f.cmd:=command;
X        waitcts(TRUE);
X        IF networkerror THEN RETURN FALSE END;
X        IF trace THEN BConOut(CON,"B") END;
X        SFRAME:=f;
X        requesttosend:=TRUE;
X        waitcts(FALSE);
X        IF networkerror THEN RETURN FALSE END;
X        IF trace THEN BConOut(CON,"C") END;
X        REPEAT
X        nrecframe;
X        Nwait(event);
X        HandleEvents();
X        IF R>retry THEN networkerror:=TRUE END;
X        IF networkerror THEN RETURN FALSE END;
X        UNTIL framewaiting AND (RFRAME.cmd=reply);
X        IF trace THEN BConOut(CON,"D") END;
X        f:=RFRAME;
```

```
X        f.rw:=5;
X        framewaiting:=FALSE;
X        sendtoremote(ack,reply,f); (* send ack for reply *)
X        IF networkerror THEN RETURN FALSE END;
X        IF trace THEN BConOut(CON,"Z") END;
X        RETURN TRUE;
XEND     getfromremote;
X
XPROCEDURE sendtoremote(type: FrameKind; command: DataKind;VAR f: frame);
XBEGIN
X        IF trace THEN BConOut(CON,"T") END;
X        f.kind:=type;
X        f.cmd:=command;
X        IF debug THEN cleartosend:=TRUE END; (* so we can send in loop *)
X        waitcts(TRUE);
X        IF trace THEN BConOut(CON,"1") END;
X        SFRAME:=f;
X        requesttosend:=TRUE;
X        waitcts(FALSE);
X        IF trace THEN BConOut(CON,"2") END;
X        IF SFRAME.kind=ack THEN cleartosend:=TRUE END;
XEND     sendtoremote;
X
XPROCEDURE senddata;
XBEGIN
X    SFRAME.seq:=C.nextframetosend;
X    SFRAME.ack:=1-C.frameexpected;
X    sendf(SFRAME);
X    IF (SFRAME.kind#ack) AND (SFRAME.kind#resetreq) THEN
X       StartTimer; (* set timer to wait for frame ack from remote host *)
X    END;
XEND     senddata;
X
X(*$P+ *)
XPROCEDURE sendf(VAR f: frame);
XBEGIN
X        sframe:=f;
X        sframe.cksum:=0;
X        IF ((sframe.cmd=rdrwabsconf) AND ((sframe.rw=0) OR (sframe.rw=2))) OR
((sframe.cmd=rdrwabsreq) AND ((sframe.rw=1) OR (sframe.rw=3))) THEN
X           sndframesize:=SIZEframe;
X           sframe.syn :=  SYN ;
X           sframe.stx :=  STX ;
X           sframe.user := USER ;
X           shortframe:=FALSE;
X        IF trace THEN BConOut(CON,":") END;
X        ELSE
X           sndframesize:=SIZEframecmd;
X           sframe.syn := SYN ;
X           sframe.stx := SOH ;
X           framecmdptr1:=ADR(sframe);
X           framecmdptr1^.user := USER ;
X           shortframe:=TRUE;
```

```
X        IF trace THEN BConOut(CON,".") END;
X        END;
X        FOR i1:=0 TO sndframesize-5 DO (* compute checksum *)
X            sframe.cksum:=sframe.cksum+CARDINAL(sframeptr^[i1])
X        END;
X        IF shortframe THEN framecmdptr1^.cksum:=sframe.cksum END;
X        FOR i1:=0 TO sndframesize-1 DO (* send frame *)
X            REPEAT
X                nrecframe;
X            UNTIL BCosStat(netdevice);
X            BConOut(netdevice,CHAR(sframeptr^[i1]));
X        END;
XEND     sendf;
X
X(*$P- *)
XPROCEDURE gettime;
XBEGIN
X        clock:=hz200 DIV 200;
X        CODE(RTS);
XEND     gettime;
X(*$P+ *)
X
XPROCEDURE getf(VAR f: frame);
XBEGIN
X        f:=rframe;
X        framebufferfull:=FALSE;
XEND     getf;
X
XPROCEDURE StartTimer;
XBEGIN
X        SuperExec(gettime);
X        timestart:=clock;  (* set to time in seconds *)
X        timer:=TRUE;                    (* test *)
X        timefortimeout:=timestart;
X        IncTime(timefortimeout,2);
XEND     StartTimer;
X
XPROCEDURE IncTime(VAR t : LONGCARD; c: CARDINAL);
XBEGIN
X        IF c<1 THEN RETURN END;
X        t:=t+LONGCARD(c);
XEND     IncTime;
X
XPROCEDURE TimeOut(): BOOLEAN;
XBEGIN
X        IF (NOT timer) THEN RETURN FALSE END;
X        SuperExec(gettime);
X        timeouttime:=clock;
X        SETREG(0,timeouttime);
X        CODE(0280H,0,0FFFFH);
X        timeouttime:=LONGCARD(REGISTER(0));
X        IF timeouttime>timefortimeout THEN
X           StartTimer;
```

```
X           RETURN TRUE;
X        END;
X        RETURN FALSE;
XEND     TimeOut;
X
XPROCEDURE Nwait(VAR e: evtype);
XBEGIN
X
X         IF requesttosend AND cleartosend THEN
X            e:=hostready;
X            requesttosend:=FALSE;
X            cleartosend:=FALSE;
X            RETURN;
X         END;
X
X         IF C.sendreset THEN
X            e:=reset;
X         END;
X
X         IF framebufferfull THEN
X           cksum:=0;
X           FOR i2:=0 TO recframesize-5 DO
X              cksum:=cksum+CARDINAL(rframeptr^[i2])
X           END;
X           IF (cksum=rframe.cksum) THEN
X              e:=framearrival;
X              INC(R);
X           ELSE
X              e:=cksumerr;
X              framebufferfull:=FALSE;
X        IF trace THEN BConOut(CON,"U") END;
X           END;
X           RETURN;
X         END;
X         nrecframe;
X         IF TimeOut() THEN
X            e:=timeout;
X            INC(R);
X         END;     (* so sorry no frame ack *)
XEND     Nwait;
X
XPROCEDURE ToHost(VAR f: frame);
XBEGIN
X        IF trace THEN BConOut(CON,"H") END;
X        IF f.kind=callreq THEN
X           framewaiting:=FALSE;
X           RETURN;
X        END;
X        IF f.kind=clearreq THEN
X           framewaiting:=FALSE;
X           RETURN;
X        END;
X        IF f.kind=diag THEN
```

```
X           framewaiting:=FALSE;
X           RETURN;
X        END;
X        IF f.kind=data THEN
X           IF f.cmd=rdmediareq THEN
X        IF trace THEN BConOut(CON,"M") END;
X              framewaiting:=FALSE;
X              nframe2.d0:=LONGCARD(MediaChange(CARDINAL(f.d0)));
X              sendtoremote(data,rdmediaconf,nframe2);
X              RETURN;
X           END;
X           IF f.cmd=rdbpbreq THEN
X        IF trace THEN BConOut(CON,"P") END;
X              framewaiting:=FALSE;
X              nframe2.d0:=LONGCARD(GetBPB(CARDINAL(f.d0)));
X              bpbptr:=ADDRESS(nframe2.d0);
X              nbpbptr:=ADR(nframe2.info);
X              FOR i:=0 TO TSIZE(BPB)-1 DO
X                  nbpbptr^[i]:=bpbptr^[i];
X              END;
X              sendtoremote(data,rdbpbconf,nframe2);
X              RETURN;
X           END;
X           IF f.cmd=rdrwabsreq THEN
X        IF trace THEN BConOut(CON,"W") END;
X              framewaiting:=FALSE;
X              nframe2.d0:=LONGCARD(RWAbs(RW(f.rw),ADR(f.info),1,f.recno,
X                                   CARDINAL(f.d0)));
X              IF (f.rw=0) OR (f.rw=2) THEN
X                 nframe2.rw:=f.rw;
X                 nframe2.info:=f.info; (* if rec get buffer to send *)
X              END;
X              sendtoremote(data,rdrwabsconf,nframe2);
X              RETURN;
X           END;
X        END;
XEND     ToHost;
X
XPROCEDURE HandleEvents();
XBEGIN
X            IF event=hostready THEN
X               event:=nothing;
X        IF trace THEN BConOut(CON,"S") END;
X               senddata;
X            END;
X
X            IF event=reset THEN
X        IF trace THEN BConOut(CON,"I") END;
X               charcount:=0;
X               R:=0;
X               gotframe:=FALSE;
X               framebufferfull:=FALSE;
X               FOR d:=0 TO 5 DO
```

```
X                   gotmediach[d]:=FALSE;
X                   gotbpb[d]:=FALSE;
X               END;
X               C.nextframetosend:=0;
X               C.frameexpected:=0;
X               cleartosend:=TRUE;
X               requesttosend:=FALSE;
X               framewaiting:=FALSE;
X               timer:=FALSE;
X               C.sendreset:=FALSE;
X               event:=nothing;
X               SFRAME.kind:=resetreq;
X               senddata;
X            END;
X
X            IF event=framearrival THEN
X               event:=nothing;
X
X               IF (rframe.kind=ack) OR (rframe.kind=resetreq) THEN
X                  framewaiting:=FALSE
X               END;
X        IF trace AND (NOT framewaiting) THEN BConOut(CON,"F") END;
X
X               IF (NOT framewaiting) THEN getf(RFRAME) END;
X               framebufferfull:=FALSE;
X
X               IF (RFRAME.ack=C.nextframetosend) OR debug THEN
X        IF trace THEN BConOut(CON,"K") END;
X                  cleartosend:=TRUE;
X                  StartTimer;
X                  R:=0;
X                  timer:=FALSE;
X                  inc(C.nextframetosend);
X               END;
X
X               IF (RFRAME.seq=C.frameexpected) OR debug THEN
X        IF trace THEN BConOut(CON,"E") END;
X                  IF RFRAME.kind#ack THEN (* try to exec command *)
X                     inc(C.frameexpected);
X                     framewaiting:=TRUE;
X                     R:=0;
X                     ToHost(RFRAME);
X                  END;
X               END;
X               IF RFRAME.kind=resetreq THEN
X        IF trace THEN BConOut(CON,"*") END;
X                  charcount:=0;
X                  gotframe:=FALSE;
X                  framebufferfull:=FALSE;
X                  C.nextframetosend:=0;
X                  C.frameexpected:=0;
X                  FOR d:=0 TO 5 DO
X                      gotmediach[d]:=FALSE;
```

```
X                       gotbpb[d]:=FALSE;
X                   END;
X                   cleartosend:=TRUE;
X                   requesttosend:=FALSE;
X                   framewaiting:=FALSE;
X                   timer:=FALSE;
X                   C.sendreset:=FALSE;
X                   event:=nothing;
X                   BConOut(CON,BEL);
X                   BConOut(CON,BEL);
X                END;
X             END;
X
X         SFRAME.seq:=C.nextframetosend;
X         SFRAME.ack:=1-C.frameexpected;
X
X         IF event=timeout THEN
X            event:=nothing;
X         IF trace THEN BConOut(CON,"R") END;
X            sendf(SFRAME);
X            framewaiting:=FALSE;
X         END;
XEND      HandleEvents;
X
XPROCEDURE       recframe;
XBEGIN
X         nrecframe;
X         Nwait(event);
X         HandleEvents();
XEND      recframe;
X
XPROCEDURE       initnetwork(port: Device);
XBEGIN
X  netdevice:=port;
X  IF Initialise(port) THEN
X
X    charcount:=0;
X    gotframe:=FALSE;
X    framebufferfull:=FALSE;
X    C.nextframetosend:=0;
X    C.frameexpected:=0;
X    FOR d:=0 TO 5 DO
X        gotmediach[d]:=FALSE;
X        gotbpb[d]:=FALSE;
X    END;
X    cleartosend:=TRUE;
X    requesttosend:=FALSE;
X    framewaiting:=FALSE;
X    timer:=FALSE;
X    C.sendreset:=FALSE;
X    event:=nothing;
X    C.networkactive:=TRUE;
X    IF NOT installed THEN
```

```
X      SuperExec(PROC(InstallVectors)) ; (* install the NETWORK *)
X      installed:=TRUE;
X    END;
X  END ;
XEND   initnetwork;
X
XPROCEDURE       networkoff;
XBEGIN
X    C.networkactive:=FALSE;
XEND     networkoff;
X
XPROCEDURE       networkon;
XBEGIN
X    C.networkactive:=TRUE;
XEND     networkon;
X
XBEGIN
XEND NETWORK.
SHAR_EOF
chmod 0600 NETWORK.MOD || echo "restore of NETWORK.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > NEWSYS.DEF &&
X
X(*              Copyright 1987 fred brooks LogicTek             *)
X(*                                                              *)
X(*                                                              *)
X(* 1.0.0  First Release                       12/8/87-FGB       *)
X(* 1.0.1  Changed to trap #9 because of possible conflict with  *)
X(*        JSM2 coroutines                     12/9/87-FGB       *)
X(*                                                              *)
X
XDEFINITION MODULE NEWSYS;
XFROM       SYSTEM       IMPORT  ADDRESS;
X
XTYPE    PROCESS         =       ADDRESS;
XCONST      TRAP         =       0A4H; (* trap 9 adr used for TRANSFER and
X                                       IOTRANSFER *)
X
XPROCEDURE  NEWPROCESS(processProc       :  PROC;
X                      workspace         :  ADDRESS;
X                      worksize          :  LONGCARD;
X                      VAR process       :  ADDRESS);
X
XPROCEDURE  TRANSFER(VAR p1,p2: ADDRESS);
X
XPROCEDURE  IOTRANSFER(VAR p1,p2: ADDRESS; device: ADDRESS);
X
XEND               NEWSYS.
SHAR_EOF
chmod 0600 NEWSYS.DEF || echo "restore of NEWSYS.DEF fails"
sed 's/^X//' << 'SHAR_EOF' > NEWSYS.MOD &&
X
X(*              Copyright 1987 fred brooks LogicTek             *)
X(*                                                              *)
```

```
X(*                                                                  *)
X(*  1.0.0 First Release                        12/8/87-FGB          *)
X(*  1.0.1 Changed to trap #9 because of possible conflict with  *)
X(*        JSM2 coroutines                      12/9/87-FGB          *)
X(*                                                                  *)
X
XIMPLEMENTATION MODULE NEWSYS; (*$S-,$T- *)
XFROM           SYSTEM   IMPORT  ADDRESS,CODE,SETREG,REGISTER,ADR;
X
XFROM    GEMDOS  IMPORT  Super,Alloc,Free;
X
XTYPE           trappointer      =       POINTER TO PROC;
X               processpointer   =       POINTER TO ADDRESS;
XTYPE           iotype   =       RECORD
X                                        p1      : processpointer;
X                                        p2      : processpointer;
X                                        device  : ADDRESS;
X                                END;
XVAR            pc,ssv                           : ADDRESS;
X               io1,io2                          : processpointer;
X               sr,function                      : CARDINAL;
X               iotranspointer                   : iotype;
X               trap                             : trappointer;
X               pr1,pr2,wsp                      : ADDRESS;
X               n                                : LONGCARD;
X               init                             : BOOLEAN;
X
XPROCEDURE  NEWPROCESS(processProc       :  PROC;
X                      workspace         :  ADDRESS;
X                      worksize          :  LONGCARD;
X                      VAR process       :  ADDRESS);
XBEGIN
X        IF NOT init THEN
X           INITPROCESSES;
X        END;
X        workspace:=workspace+ADDRESS(worksize);
X        SETREG(6,ADDRESS(processProc));
X        SETREG(8,workspace);
X        CODE(2106H);    (* move.l d6,-(a0) PC *)
X        CODE(313cH,0300H);      (* move.w $0300,-(a0)   CCR  *)
X        CODE(48e0H,0fffeH);     (* movem.l d0-d7/a0-a6,-(a0) *)
X        process:=REGISTER(8);
XEND        NEWPROCESS;
X
XPROCEDURE  TRANSFER(VAR p1,p2: ADDRESS);
XBEGIN      (* pass p1 and p2 as the location of these variables *)
X        IF NOT init THEN
X           INITPROCESSES;
X        END;
X           SETREG(0,ADR(p2));
X           CODE(2f00H); (* move.l d0,-(sp) *)
X           SETREG(0,ADR(p1));
X           CODE(2f00H); (* move.l d0,-(sp) *)
```

```
X           CODE(3f3cH,1); (* move.w #1,-(sp) *)
X           CODE(4e49H); (* trap #9 *)
X           CODE(0dffcH,0,10); (* add.l #10,sp *)
XEND        TRANSFER;
X
XPROCEDURE  IOTRANSFER(VAR p1,p2: ADDRESS; device: ADDRESS);
XBEGIN      (* pass p1 and p2 as the location of these variables *)
X        IF NOT init THEN
X           INITPROCESSES;
X        END;
X           SETREG(0,device);
X           CODE(2f00H); (* move.l d0,-(sp) *)
X           SETREG(0,ADR(p2));
X           CODE(2f00H); (* move.l d0,-(sp) *)
X           SETREG(0,ADR(p1));
X           CODE(2f00H); (* move.l d0,-(sp) *)
X           CODE(3f3cH,2); (* move.w #2,-(sp) *)
X           CODE(4e49H); (* trap #9 *)
X           CODE(0dffcH,0,14); (* add.l #14,sp *)
XEND        IOTRANSFER;
X
X(*$P- *)
XPROCEDURE       PTRAP;
XBEGIN
X        CODE(043374B,2700H); (* disable ints *)
X        CODE(48e7H,0fffeH);   (* save regs movem  *)
X        CODE(306fH,60); (* move.w 60(a7),a0 get sr *)
X        sr:=CARDINAL(REGISTER(8));
X        IF sr>3fffH THEN         (* called from supermode, not valid *)
X           CODE(4cdfH,7fffH);    (* restore regs movem *)
X           CODE(4e73H);          (* rte go back to where we came from *)
X        END;
X
X        CODE(4e69H);        (* move.l usp,a1 *)
X        CODE(3069H,0);      (* move.w 0(a1),a0 *)
X        function:=CARDINAL(REGISTER(8));
X        CODE(4e69H);        (* move.l usp,a1 *)
X        CODE(2069H,2);      (* move.l 2(a1),a0 *)
X        iotranspointer.p1:=REGISTER(8);
X        CODE(4e69H);        (* move.l usp,a1 *)
X        CODE(2069H,6);      (* move.l 6(a1),a0 *)
X        iotranspointer.p2:=REGISTER(8);
X        CODE(4e69H);        (* move.l usp,a1 *)
X        CODE(2069H,10);     (* move.l 10(a1),a0 *)
X        iotranspointer.device:=REGISTER(8);
X
X        CASE function OF
X             1  : CODE(4e68H); (* move.l usp,a0 TRANSFER *) (* SAVE *)
X                  CODE(0dffcH,0,42H); (* add.l #66,sp *)
X                  CODE(2127H); (* move.l -(sp),-(a0) D0 *)
X                  CODE(2127H); (* move.l -(sp),-(a0) D1 *)
X                  CODE(2127H); (* move.l -(sp),-(a0) D2 *)
X                  CODE(2127H); (* move.l -(sp),-(a0) D3 *)
```

```
X                  CODE(2127H); (* move.l -(sp),-(a0) D4 *)
X                  CODE(2127H); (* move.l -(sp),-(a0) D5 *)
X                  CODE(2127H); (* move.l -(sp),-(a0) D6 *)
X                  CODE(2127H); (* move.l -(sp),-(a0) D7 *)
X                  CODE(2127H); (* move.l -(sp),-(a0) A0 *)
X                  CODE(2127H); (* move.l -(sp),-(a0) A1 *)
X                  CODE(2127H); (* move.l -(sp),-(a0) A2 *)
X                  CODE(2127H); (* move.l -(sp),-(a0) A3 *)
X                  CODE(2127H); (* move.l -(sp),-(a0) A4 *)
X                  CODE(2127H); (* move.l -(sp),-(a0) A5 *)
X                  CODE(2127H); (* move.l -(sp),-(a0) A6 *)
X                  CODE(3127H); (* move.w -(sp),-(a0) SR *)
X                  CODE(2127H); (* move.l -(sp),-(a0) PC *)
X                  iotranspointer.p1^:=REGISTER(8); (* set p1 to process *)
X
X                  SETREG(8,iotranspointer.p2^); (* load p2 to a0 RESTORE *)
X                  CODE(2ed8H); (* move.l (a0)+,(sp)+ D0 *)
X                  CODE(2ed8H); (* move.l (a0)+,(sp)+ D1 *)
X                  CODE(2ed8H); (* move.l (a0)+,(sp)+ D2 *)
X                  CODE(2ed8H); (* move.l (a0)+,(sp)+ D3 *)
X                  CODE(2ed8H); (* move.l (a0)+,(sp)+ D4 *)
X                  CODE(2ed8H); (* move.l (a0)+,(sp)+ D5 *)
X                  CODE(2ed8H); (* move.l (a0)+,(sp)+ D6 *)
X                  CODE(2ed8H); (* move.l (a0)+,(sp)+ D7 *)
X                  CODE(2ed8H); (* move.l (a0)+,(sp)+ A0 *)
X                  CODE(2ed8H); (* move.l (a0)+,(sp)+ A1 *)
X                  CODE(2ed8H); (* move.l (a0)+,(sp)+ A2 *)
X                  CODE(2ed8H); (* move.l (a0)+,(sp)+ A3 *)
X                  CODE(2ed8H); (* move.l (a0)+,(sp)+ A4 *)
X                  CODE(2ed8H); (* move.l (a0)+,(sp)+ A5 *)
X                  CODE(2ed8H); (* move.l (a0)+,(sp)+ A6 *)
X                  CODE(3ed8H); (* move.w (a0)+,(sp)+ SR *)
X                  CODE(2ed8H); (* move.l (a0)+,(sp)+ PC *)
X                  CODE(9ffcH,0,42H); (* sub.l #66,sp *)
X                  CODE(4e60H); (* move.l a0,usp *)
X                  CODE(4cdfH,7fffH);    (* restore regs movem *)
X                  CODE(4e73H); |        (* rte *)
X
X              2  : CODE(4e68H); (* move.l usp,a0 IOTRANSFER *) (* SAVE *)
X                   CODE(0dffcH,0,42H); (* add.l #66,sp *)
X                   CODE(2127H); (* move.l -(sp),-(a0) D0 *)
X                   CODE(2127H); (* move.l -(sp),-(a0) D1 *)
X                   CODE(2127H); (* move.l -(sp),-(a0) D2 *)
X                   CODE(2127H); (* move.l -(sp),-(a0) D3 *)
X                   CODE(2127H); (* move.l -(sp),-(a0) D4 *)
X                   CODE(2127H); (* move.l -(sp),-(a0) D5 *)
X                   CODE(2127H); (* move.l -(sp),-(a0) D6 *)
X                   CODE(2127H); (* move.l -(sp),-(a0) D7 *)
X                   CODE(2127H); (* move.l -(sp),-(a0) A0 *)
X                   CODE(2127H); (* move.l -(sp),-(a0) A1 *)
X                   CODE(2127H); (* move.l -(sp),-(a0) A2 *)
X                   CODE(2127H); (* move.l -(sp),-(a0) A3 *)
X                   CODE(2127H); (* move.l -(sp),-(a0) A4 *)
```

```
X                 CODE(2127H); (* move.l -(sp),-(a0) A5 *)
X                 CODE(2127H); (* move.l -(sp),-(a0) A6 *)
X                 CODE(3127H); (* move.w -(sp),-(a0) SR *)
X                 CODE(2127H); (* move.l -(sp),-(a0) PC *)
X                 iotranspointer.p1^:=REGISTER(8); (* set p1 to process *)
X                 io1:=iotranspointer.p1;
X
X                 io2:=iotranspointer.p2;
X                 SETREG(8,iotranspointer.p2^); (* load p2 to a0 RESTORE *)
X                 CODE(2ed8H); (* move.l (a0)+,(sp)+ D0 *)
X                 CODE(2ed8H); (* move.l (a0)+,(sp)+ D1 *)
X                 CODE(2ed8H); (* move.l (a0)+,(sp)+ D2 *)
X                 CODE(2ed8H); (* move.l (a0)+,(sp)+ D3 *)
X                 CODE(2ed8H); (* move.l (a0)+,(sp)+ D4 *)
X                 CODE(2ed8H); (* move.l (a0)+,(sp)+ D5 *)
X                 CODE(2ed8H); (* move.l (a0)+,(sp)+ D6 *)
X                 CODE(2ed8H); (* move.l (a0)+,(sp)+ D7 *)
X                 CODE(2ed8H); (* move.l (a0)+,(sp)+ A0 *)
X                 CODE(2ed8H); (* move.l (a0)+,(sp)+ A1 *)
X                 CODE(2ed8H); (* move.l (a0)+,(sp)+ A2 *)
X                 CODE(2ed8H); (* move.l (a0)+,(sp)+ A3 *)
X                 CODE(2ed8H); (* move.l (a0)+,(sp)+ A4 *)
X                 CODE(2ed8H); (* move.l (a0)+,(sp)+ A5 *)
X                 CODE(2ed8H); (* move.l (a0)+,(sp)+ A6 *)
X                 CODE(3ed8H); (* move.w (a0)+,(sp)+ SR *)
X                 CODE(2ed8H); (* move.l (a0)+,(sp)+ PC *)
X                 CODE(9ffcH,0,42H); (* sub.l #66,sp *)
X                 CODE(4e60H); (* move.l a0,usp *)
X                 trap:=trappointer(iotranspointer.device); (* TRAP ADR *)
X                 trap^:=ITRAP; (* set trap to IOTRANSFER int code *)
X                 CODE(4cdfH,7fffH);    (* restore regs movem *)
X                 CODE(4e73H); |        (* rte *)
X         END;
X
X         CODE(4cdfH,7fffH);    (* restore regs movem *)
X         CODE(4e73H);          (* rte *)
XEND              PTRAP;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE        ITRAP;
XBEGIN
X         CODE(043374B,2700H); (* disable ints *)
X         CODE(48e7H,0fffeH);   (* save regs movem  *)
X         CODE(4e68H); (* move.l usp,a0 TRANSFER *) (* SAVE *)
X         CODE(0dffcH,0,42H); (* add.l #66,sp *)
X         CODE(2127H); (* move.l -(sp),-(a0) D0 *)
X         CODE(2127H); (* move.l -(sp),-(a0) D1 *)
X         CODE(2127H); (* move.l -(sp),-(a0) D2 *)
X         CODE(2127H); (* move.l -(sp),-(a0) D3 *)
X         CODE(2127H); (* move.l -(sp),-(a0) D4 *)
X         CODE(2127H); (* move.l -(sp),-(a0) D5 *)
X         CODE(2127H); (* move.l -(sp),-(a0) D6 *)
```

```
X        CODE(2127H); (* move.l -(sp),-(a0) D7 *)
X        CODE(2127H); (* move.l -(sp),-(a0) A0 *)
X        CODE(2127H); (* move.l -(sp),-(a0) A1 *)
X        CODE(2127H); (* move.l -(sp),-(a0) A2 *)
X        CODE(2127H); (* move.l -(sp),-(a0) A3 *)
X        CODE(2127H); (* move.l -(sp),-(a0) A4 *)
X        CODE(2127H); (* move.l -(sp),-(a0) A5 *)
X        CODE(2127H); (* move.l -(sp),-(a0) A6 *)
X        CODE(3127H); (* move.w -(sp),-(a0) SR *)
X        CODE(2127H); (* move.l -(sp),-(a0) PC *)
X        io2^:=REGISTER(8); (* set interrupted process to process *)
X
X        SETREG(8,io1^); (* load iotransfer process to a0 RESTORE *)
X        CODE(2ed8H); (* move.l (a0)+,(sp)+ D0 *)
X        CODE(2ed8H); (* move.l (a0)+,(sp)+ D1 *)
X        CODE(2ed8H); (* move.l (a0)+,(sp)+ D2 *)
X        CODE(2ed8H); (* move.l (a0)+,(sp)+ D3 *)
X        CODE(2ed8H); (* move.l (a0)+,(sp)+ D4 *)
X        CODE(2ed8H); (* move.l (a0)+,(sp)+ D5 *)
X        CODE(2ed8H); (* move.l (a0)+,(sp)+ D6 *)
X        CODE(2ed8H); (* move.l (a0)+,(sp)+ D7 *)
X        CODE(2ed8H); (* move.l (a0)+,(sp)+ A0 *)
X        CODE(2ed8H); (* move.l (a0)+,(sp)+ A1 *)
X        CODE(2ed8H); (* move.l (a0)+,(sp)+ A2 *)
X        CODE(2ed8H); (* move.l (a0)+,(sp)+ A3 *)
X        CODE(2ed8H); (* move.l (a0)+,(sp)+ A4 *)
X        CODE(2ed8H); (* move.l (a0)+,(sp)+ A5 *)
X        CODE(2ed8H); (* move.l (a0)+,(sp)+ A6 *)
X        CODE(3ed8H); (* move.w (a0)+,(sp)+ SR *)
X        CODE(2ed8H); (* move.l (a0)+,(sp)+ PC *)
X        CODE(9ffcH,0,42H); (* sub.l #66,sp *)
X        CODE(4e60H); (* move.l a0,usp *)
X        CODE(4cdfH,7fffH);    (* restore regs movem *)
X        CODE(4e73H);         (* rte *)
XEND             ITRAP;
X(*$P+ *)
X
XPROCEDURE  INITPROCESSES;
XBEGIN
X        ssv:=0;
X        Super(ssv);
X        trap:=trappointer(TRAP);
X        trap^:=PTRAP;
X        Super(ssv);
X        init:=TRUE;
XEND        INITPROCESSES;
X
XBEGIN
XEND               NEWSYS.
SHAR_EOF
chmod 0600 NEWSYS.MOD || echo "restore of NEWSYS.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > PASSWD.MOD &&
XMODULE  PASSWD;
```

```
XFROM    Terminal   IMPORT  ReadString,WriteString,WriteLn;
XFROM    TermBase   IMPORT  DoWrite,WriteProc;
XFROM    Strings    IMPORT  String,Compare,CompareResults,Pos,Copy,Concat,
X                           Length;
XFROM    GEMDOS     IMPORT  OldTerm,Rename,Delete,Open,Close;
XIMPORT  TextIO;
XFROM    Streams    IMPORT  StreamKinds,CloseStream;
XFROM    GEMX       IMPORT  BasePageAddress;
XFROM    LOGIN      IMPORT  crypt;
XCONST   USERS = 16;
XTYPE    sString =       ARRAY [0..13] OF CHAR;
X        lString =       ARRAY [0..64] OF CHAR;
XVAR     user,password,prg       :       ARRAY [0..USERS-1] OF sString;
X        path,com                :       ARRAY [0..USERS-1] OF lString;
X        input,clipath           :       String;
X        black                   :       sString;
X        i,index,spos,epos       :       CARDINAL;
X        userfound,ok            :       BOOLEAN;
X        normal,noecho           :       WriteProc;
X        result                  :       INTEGER;
X
XPROCEDURE       NoEcho(char: CHAR);
XBEGIN
XEND             NoEcho;
X
XBEGIN
X        Open("passwd",0,result);
X        IF result>0 THEN
X           ok:=Close(result);
X        ELSE
X           OldTerm;
X        END;
X        TextIO.SetDefaultIO("PASSWD",READ,result);
X        normal:=DoWrite;
X        noecho:=NoEcho;
X        FOR i:=0 TO USERS-1 DO               (* read in userfile *)
X                TextIO.ReadString(input);
X                IF input[0]#0c THEN
X                spos:=0;
X                ok:=Pos(input,":",spos,epos);
X                Copy(input,spos,epos-spos,user[i]);
X                spos:=epos+1;
X                ok:=Pos(input,":",spos,epos);
X                Copy(input,spos,epos-spos,password[i]);
X                spos:=epos+1;
X                ok:=Pos(input,":",spos,epos);
X                Copy(input,spos,epos-spos,path[i]);
X                spos:=epos+1;
X                ok:=Pos(input,":",spos,epos);
X                Copy(input,spos,epos-spos,prg[i]);
X                spos:=epos+1;
X                Copy(input,spos,Length(input)-spos,com[i]);
X                END;
```

```
X        END; (* for *)
X        CloseStream(TextIO.in,result);
X        CloseStream(TextIO.out,result);
X        LOOP
X           userfound:=FALSE;
X           FOR i:=0 TO CARDINAL(BasePageAddress^.CmdLine[0]) DO
X               input[i]:=BasePageAddress^.CmdLine[i+1]; (* read in command line
*)
X           END;
X           input[i+1]:=0c;
X           IF input[0]=0c THEN OldTerm END; (* no user exit *)
X           FOR i:=0 TO USERS-1 DO
X               IF Compare(input,user[i])=Equal THEN
X                  userfound:=TRUE;
X                  index:=i;
X               END;
X           END;
X           IF (password[index][0]#0c) OR (NOT userfound) THEN
X           WriteString("Old Password: ");
X           DoWrite:=noecho;
X           ReadString(input);
X           DoWrite:=normal;
X           WriteLn;
SHAR_EOF
echo "End of part 5, continue with part 6"
echo "6" > s2_seq_.tmp
exit 0


From -7845006050703965177
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,b78d5fc4f2c1f2bd,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-12-27 13:35:04 PST
Path: utzoo!attcan!uunet!husc6!bloom-beacon!tut.cis.ohio-state.edu!rutgers!ucsd!
ucbvax!agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i086: mx2v230 -- Multitasking kernel with utilities  part06/08
Keywords: shar, Modula-2
Message-ID: <5872@saturn.ucsc.edu>
Date: 27 Dec 88 21:35:04 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1031
Approved: koreth@ssyx.ucsc.edu

Submitted-by: madsen@sask.usask.ca (Jorgen Madsen)
Posting-number: Volume 1, Issue 86
Archive-name: mx2v230/part06

#!/bin/sh
# this is part 6 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
```

```
# file PASSWD.MOD continued
#
CurArch=6
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> PASSWD.MOD
X           black:=user[index];
X           crypt(input,black);
X           IF (Compare(black,password[index])=Equal) AND userfound THEN
X              EXIT;
X           END;
X           END;
X           IF (password[index][0]=0c) AND userfound THEN EXIT END;
X           WriteString("Incorrect Password not changed!");
X           WriteLn;
X           OldTerm;
X        END; (* loop *)
X        REPEAT
X        WriteString("New Password: ");
X        DoWrite:=noecho;
X        ReadString(password[index]);
X        DoWrite:=normal;
X        WriteLn;
X        IF Length(password[index])<4 THEN
X           WriteString("Password too short. Please use a longer password.");
X           WriteLn;
X        END;
X        UNTIL (Length(password[index])>=4);
X        black:=user[index];
X        crypt(password[index],black);
X        password[index]:=black;
X        ok:=Delete("passwd.old");
X        Rename("passwd","passwd.old");
X        ok:=Delete("passwd");
X        TextIO.SetDefaultIO("PASSWD",READWRITE,result);
X        FOR i:=0 TO USERS-1 DO               (* Write passwd file *)
X            IF user[i][0]#0c THEN
X                TextIO.WriteString(user[i]);
X                TextIO.Write(':');
X                TextIO.WriteString(password[i]);
X                TextIO.Write(':');
X                TextIO.WriteString(path[i]);
X                TextIO.Write(':');
X                TextIO.WriteString(prg[i]);
X                TextIO.Write(':');
X                TextIO.WriteString(com[i]);
```

```
X               TextIO.WriteLn;
X           END;
X       END; (* for *)
X       CloseStream(TextIO.in,result);
X       CloseStream(TextIO.out,result);
X       OldTerm;
XEND    PASSWD.
X
SHAR_EOF
chmod 0600 PASSWD.MOD || echo "restore of PASSWD.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > PIPE.DEF &&
X
X(*              Copyright 1987 fred brooks LogicTek            *)
X(*                                                             *)
X(*                                                             *)
X(*   First Release                      12/8/87-FGB            *)
X(* Correct Bad error in PipeOpen causing pipe to be closed if  *)
X(* this routine is called               12/14/87-FGB           *)
X(*                                                             *)
X
XDEFINITION MODULE PIPE;
XFROM       SYSTEM       IMPORT  BYTE,WORD,LONGWORD;
X
XPROCEDURE OpenPipe(pipeName: ARRAY OF CHAR): LONGCARD; (* return pipe id *)
XPROCEDURE IsReadable(pipeid: LONGCARD): BOOLEAN;
XPROCEDURE IsWriteable(pipeid: LONGCARD): BOOLEAN;
XPROCEDURE ClosePipe(pipeid: LONGCARD);
XPROCEDURE PipeOpen(pipeid: LONGCARD): BOOLEAN;
X
XPROCEDURE PWriteByte(pipeid: LONGCARD; byte: BYTE): BOOLEAN;
XPROCEDURE PWriteWord(pipeid: LONGCARD; word: WORD): BOOLEAN;
XPROCEDURE PWriteLongWord(pipeid: LONGCARD; longword: LONGWORD): BOOLEAN;
X
XPROCEDURE PReadByte(pipeid: LONGCARD; VAR byte: BYTE): BOOLEAN;
XPROCEDURE PReadWord(pipeid: LONGCARD; VAR word: WORD): BOOLEAN;
XPROCEDURE PReadLongWord(pipeid: LONGCARD; VAR longword: LONGWORD): BOOLEAN;
XEND              PIPE.
SHAR_EOF
chmod 0600 PIPE.DEF || echo "restore of PIPE.DEF fails"
sed 's/^X//' << 'SHAR_EOF' > PIPE.MOD &&
X
X(*              Copyright 1987 fred brooks LogicTek            *)
X(*                                                             *)
X(*                                                             *)
X(*   First Release                      12/8/87-FGB            *)
X(* Correct Bad error in PipeOpen causing pipe to be closed if  *)
X(* this routine is called               12/14/87               *)
X(*                                                             *)
X
X(*$S-,$T- *)
XIMPLEMENTATION MODULE PIPE [7];
XFROM    SYSTEM    IMPORT   BYTE,WORD,LONGWORD,ADDRESS,TSIZE,ADR;
XFROM    ATOMIC    IMPORT   pipetype,sysvariable,PIPE,pipeptr,buflength;
```

```
XFROM    SYSCALL   IMPORT   SysVar;
XFROM    Storage   IMPORT   ALLOCATE,DEALLOCATE;
XFROM    Strings   IMPORT   Assign,Compare,CompareResults;
XFROM    TextIO    IMPORT   WriteString,WriteLn,ReadLn;
XCONST             bufend = buflength-1;
XVAR               pipe1,pipe          :       pipeptr;
X                  junk                :       BYTE;
X                  pipespointer        :       POINTER TO pipetype;
X                  sysvar              :       sysvariable;
X                  I                   :       CARDINAL;
X                  found               :       BOOLEAN;
X                b                  :         POINTER TO ARRAY [0..1] OF BYTE;
X                d                  :         CARDINAL;
X                b1                 :         POINTER TO ARRAY [0..3] OF BYTE;
X             newpipeptr            :     pipeptr;
X
XPROCEDURE       PWriteByte(p: LONGCARD; b: BYTE): BOOLEAN;
XBEGIN
X        IF NOT PipeOpen(p) THEN RETURN FALSE END;
X        pipe:=pipeptr(p);
X        IF (NOT (pipe^.cnt < buflength)) THEN RETURN FALSE END;
X        deposit(pipe,b);
X        RETURN TRUE;
XEND     PWriteByte;
X
XPROCEDURE       PReadByte(p: LONGCARD; VAR b: BYTE): BOOLEAN;
XBEGIN
X        IF NOT PipeOpen(p) THEN RETURN FALSE END;
X        pipe:=pipeptr(p);
X        IF (NOT (pipe^.cnt > 0)) THEN RETURN FALSE END;
X        b:=withdraw(pipe);
X        RETURN TRUE;
XEND     PReadByte;
X
XPROCEDURE       PWriteWord(p: LONGCARD; w: WORD): BOOLEAN;
XBEGIN
X        IF NOT PipeOpen(p) THEN RETURN FALSE END;
X        pipe:=pipeptr(p);
X        IF (NOT (pipe^.cnt < buflength-1)) THEN RETURN FALSE END;
X        b:=ADR(w);
X        FOR d:=0 TO 1 DO
X            deposit(pipe,b^[d]);
X        END;
X        RETURN TRUE;
XEND     PWriteWord;
X
XPROCEDURE       PReadWord(p: LONGCARD; VAR w: WORD): BOOLEAN;
XBEGIN
X        IF NOT PipeOpen(p) THEN RETURN FALSE END;
X        pipe:=pipeptr(p);
X        IF (NOT (pipe^.cnt > 1)) THEN RETURN FALSE END;
X        b:=ADR(w);
X        FOR d:=0 TO 1 DO
```

```
X            b^[d]:=withdraw(pipe);
X        END;
X        RETURN TRUE;
XEND     PReadWord;
X
XPROCEDURE       PWriteLongWord(p: LONGCARD; lw: LONGWORD): BOOLEAN;
XBEGIN
X        IF NOT PipeOpen(p) THEN RETURN FALSE END;
X        pipe:=pipeptr(p);
X        IF (NOT (pipe^.cnt < buflength-3)) THEN RETURN FALSE END;
X        b1:=ADR(lw);
X        FOR d:=0 TO 3 DO
X            deposit(pipe,b1^[d]);
X        END;
X        RETURN TRUE;
XEND     PWriteLongWord;
X
XPROCEDURE       PReadLongWord(p: LONGCARD; VAR lw: LONGWORD): BOOLEAN;
XBEGIN
X        IF NOT PipeOpen(p) THEN RETURN FALSE END;
X        pipe:=pipeptr(p);
X        IF (NOT (pipe^.cnt > 3)) THEN RETURN FALSE END;
X        b1:=ADR(lw);
X        FOR d:=0 TO 3 DO
X            b1^[d]:=withdraw(pipe);
X        END;
X        RETURN TRUE;
XEND     PReadLongWord;
X
XPROCEDURE      deposit(VAR tpipe: pipeptr; byte: BYTE);
XBEGIN
X        IF tpipe^.cnt < buflength THEN
X           INC(tpipe^.cnt);
X        ELSE
X           (* pipe full *)
X           RETURN;
X        END;
X        tpipe^.buf[tpipe^.bufhead]:=byte;
X        IF tpipe^.bufhead=bufend THEN
X           tpipe^.bufhead:=0;
X        ELSE
X           INC(tpipe^.bufhead);
X        END;
XEND     deposit;
X
XPROCEDURE      withdraw(VAR tpipe: pipeptr): BYTE;
XBEGIN
X        IF tpipe^.cnt > 0 THEN
X           DEC(tpipe^.cnt);
X        ELSE
X           (* pipe EMPTY *)
X           RETURN BYTE(0);
X        END;
```

```
X        IF tpipe^.buftail = bufend THEN
X           tpipe^.buftail:=0;
X        ELSE
X           INC(tpipe^.buftail);
X        END;
X        RETURN tpipe^.buf[tpipe^.buftail];
XEND     withdraw;
X
XPROCEDURE       OpenPipe(pipeName: ARRAY OF CHAR): LONGCARD;
XBEGIN
X        SysVar(sysvar);
X        pipespointer:=ADDRESS(sysvar.pipes);
X
X        I:=0;   (* look for pipe name in system pipe list *)
X        found:=FALSE;
X        WHILE (I#32) AND (pipespointer^[I]#NIL) DO
X              newpipeptr:=pipespointer^[I];
X              INC(I);
X              IF Compare(pipeName,newpipeptr^.pipename)=Equal THEN
X                 I:=32;
X                 found:=TRUE;
X              END;
X        END;
X
X        IF (NOT found) THEN
X           ALLOCATE(newpipeptr,LONGCARD(TSIZE(PIPE)));
X           Assign(newpipeptr^.pipename,pipeName);
X           newpipeptr^.bufhead:=0;
X           newpipeptr^.buftail:=bufend;
X           newpipeptr^.cnt:=0;
X           newpipeptr^.bufsize:=buflength;
X
X           (* put address of pipe in system list *)
X           I:=0;
X           LOOP
X                IF I=32 THEN HALT END;
X                IF pipespointer^[I]=NIL THEN
X                   pipespointer^[I]:=newpipeptr;
X                   EXIT;
X                END;
X                INC(I);
X           END;
X        END;
X        RETURN LONGCARD(newpipeptr);
XEND     OpenPipe;
X
XPROCEDURE       ClosePipe(p: LONGCARD);
XVAR             pipe          :     pipeptr;
XBEGIN
X           pipe:=pipeptr(p);
X           SysVar(sysvar);
X           pipespointer:=ADDRESS(sysvar.pipes);
X
```

```
X           (* find address of pipe in system list *)
X           found:=FALSE;
X           I:=0;
X           LOOP
X                IF I=32 THEN EXIT END;
X                IF pipespointer^[I]=pipe THEN
X                   pipespointer^[I]:=NIL;
X                   found:=TRUE;
X                   EXIT;
X                END;
X                INC(I);
X           END;
X
X        IF found THEN
X           DEALLOCATE(pipe,LONGCARD(TSIZE(PIPE)));
X        END;
XEND     ClosePipe;
X
XPROCEDURE       PipeOpen(p: LONGCARD): BOOLEAN;
XVAR             pipe          :     pipeptr;
XBEGIN
X           pipe:=pipeptr(p);
X           SysVar(sysvar);
X           pipespointer:=ADDRESS(sysvar.pipes);
X
X           (* find address of pipe in system list *)
X           I:=0;
X           LOOP
X                IF I=32 THEN RETURN FALSE END;
X                IF pipespointer^[I]=pipe THEN
X                   RETURN TRUE;
X                END;
X                INC(I);
X           END;
XEND     PipeOpen;
X
XPROCEDURE       IsReadable(p: LONGCARD): BOOLEAN;
XBEGIN
X        pipe:=pipeptr(p);
X        IF pipe^.cnt > 0 THEN
X           RETURN TRUE;
X        ELSE
X           RETURN FALSE;
X        END;
XEND     IsReadable;
X
XPROCEDURE       IsWriteable(p: LONGCARD): BOOLEAN;
XBEGIN
X        pipe:=pipeptr(p);
X        IF pipe^.cnt < buflength THEN
X           RETURN TRUE;
X        ELSE
X           RETURN FALSE;
```

```
X        END;
XEND     IsWriteable;
X
XBEGIN
XEND                    PIPE.
SHAR_EOF
chmod 0600 PIPE.MOD || echo "restore of PIPE.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > PS.MOD &&
X(*$T-,$S-,$A+ *)
XMODULE  ps;
XFROM    SYSCALL IMPORT  SysVar,SysDes;
XFROM    GEMDOS  IMPORT  OldTerm,GetTime,GetDate,Alloc,Write;
XFROM    GEMX    IMPORT  BasePageAddress;
XFROM    SYSTEM  IMPORT  CODE,ADDRESS,ADR;
XFROM    Terminal        IMPORT  WriteString,WriteLn;
XFROM    XBIOS           IMPORT  SuperExec;
XFROM    ATOMIC          IMPORT  VERSION,SIGNAL,sysvariable,devicetype;
XFROM    Strings         IMPORT  String,Concat,Copy,Length;
XFROM    Conversions     IMPORT  ConvertToString;
XFROM    M2Conversions   IMPORT  ConvertCardinal,ConvertReal;
XFROM    BitStuff        IMPORT  WAnd,WShr;
XFROM    SCANNER         IMPORT  scinit,nxparm,ltext,etext,bkparm;
XFROM    TermBase        IMPORT  DoWrite,WriteProc;
X
XCONST
X  SPACES   = "                                                              ";
X  TITLE1 = "       ";
X  TITLE2 = " Copyright LogicTek 1987,1988 Fred Brooks ";
X  PS1      =
X"  current            system    user      total CPU    total context  CRON";
X  PS2      =
X"  time     date      memory    memory    time         switches       timer";
X  PS3      =
X"  Name           PID PPID Status Pri Init Time          CPU time    PORT";
XVAR     temp,pstemp             :       String;
X        time,date               :       CARDINAL;
X        membot [432H]           :       LONGCARD;
X        memtop [436H]           :       LONGCARD;
X        hz200  [4baH]           :       LONGCARD;
X        s0,currentprocess       :       SIGNAL;
X        sysvar                  :       sysvariable;
X        done,dmode,stmode       :       BOOLEAN;
X        temphz200,topusermem,botusermem,usermemleft,memreserve
X                                :       LONGCARD;
X
X
XPROCEDURE       FileWrite(char: CHAR);
XVAR     writecount      :       LONGCARD;
XBEGIN
X        writecount:=1;
X        Write(1,writecount,ADR(char));
XEND             FileWrite;
X
```

```
X(*$P- *)
XPROCEDURE       showmemory;
XVAR             memsize :       ADDRESS;
XBEGIN
X        topusermem:=memtop;
X        botusermem:=membot;
X        temphz200:=hz200;
X        Alloc(LONGCARD(-1),memsize);
X        usermemleft:=LONGCARD(memsize);
X        CODE(4e75H); (* rts *)
XEND     showmemory;
X(*$P+ *)
X
X(*$P- *)
XPROCEDURE       gettick;
XBEGIN
X        temphz200:=hz200;
X        CODE(4e75H); (* rts *)
XEND     gettick;
X(*$P+ *)
X
X
XPROCEDURE       FormatTD(time: CARDINAL; date: CARDINAL);
XBEGIN
X           ConvertCardinal(CARDINAL(WShr(WAnd(time,63488),11)),2,temp);
X           WriteString(temp); (* hour *)
X           WriteString(":");
X
X           ConvertCardinal(CARDINAL(WShr(WAnd(time,2016),5)),2,temp);
X           WriteString(temp); (* minute *)
X           WriteString(":");
X
X           ConvertCardinal(2*CARDINAL(WAnd(time,31)),2,temp);
X           WriteString(temp); (* second *)
X           WriteString(" ");
X
X           ConvertCardinal(CARDINAL(WShr(WAnd(date,480),5)),2,temp);
X           WriteString(temp); (* month *)
X           WriteString("/");
X
X           ConvertCardinal(CARDINAL(WAnd(date,31)),2,temp);
X           WriteString(temp); (* day *)
X           WriteString("/");
X
X           ConvertCardinal(80+CARDINAL(WShr(WAnd(date,65024),9)),2,temp);
X           WriteString(temp); (* year *)
X           WriteString("  ");
XEND             FormatTD;
X
XPROCEDURE       showtime(tick: LONGCARD);
XVAR             sec             :        REAL;
X                min,hour        :        LONGCARD;
XBEGIN
```

```
X           sec:=FLOAT(tick);
X           sec:=sec/200.0;
X           min:=TRUNCD(sec) DIV 60;
X           hour:=min DIV 60;
X           min:=min-(hour*60);
X           ConvertToString(hour,10,FALSE,temp,done);
X           scinit(ADR(temp),SIZE(temp));
X           nxparm;
X           ltext(ADR(temp),SIZE(temp));
X           WriteString(temp);
X           WriteString(":");
X           ConvertToString(min,10,FALSE,temp,done);
X           scinit(ADR(temp),SIZE(temp));
X           nxparm;
X           ltext(ADR(temp),SIZE(temp));
X           WriteString(temp);
X           WriteString(":");
X           sec:=sec-FLOAT(min*60)-FLOAT(hour*3600);
X           ConvertReal(sec,11,3,temp);
X           scinit(ADR(temp),SIZE(temp));
X           nxparm;
X           ltext(ADR(temp),SIZE(temp));
X           WriteString(temp);
XEND            showtime;
X
X(* return time in seconds *)
XPROCEDURE      converttime(time: CARDINAL): LONGCARD;
XVAR               h,m,s          :          LONGCARD;
XBEGIN
X        h:=LONGCARD(WShr(WAnd(time,63488),11));  (* hours *)
X        m:=LONGCARD(WShr(WAnd(time,2016),5));    (* minutes *)
X        s:=2*LONGCARD(WAnd(time,31));            (* seconds *)
X        RETURN s+(m*60)+(h*3600);
XEND            converttime;
X
X
XBEGIN
X           DoWrite:=WriteProc(FileWrite);
X           IF BasePageAddress^.CmdLine[2]='z' THEN stmode:=TRUE END;
X           SysDes(currentprocess);
X           SysVar(sysvar);
X           SuperExec(showmemory); (* return memsize *)
X           WriteLn;
X           WriteString(TITLE1);
X           WriteString(VERSION);
X           WriteString(TITLE2);
X           WriteLn;
X           GetTime(time);
X           GetDate(date);
X           WriteString(PS1);
X           WriteLn;
X           WriteString(PS2);
X           WriteLn;
```

```
X            WriteString("  ");
X            FormatTD(time,date);
X            ConvertToString(topusermem,10,FALSE,temp,done);
X            Copy(SPACES,0,10-Length(temp),pstemp);
X            Concat(temp,pstemp,temp);
X            WriteString(temp);
X            ConvertToString(usermemleft,10,FALSE,temp,done);
X            Copy(SPACES,0,10-Length(temp),pstemp);
X            Concat(temp,pstemp,temp);
X            WriteString(temp);
X            showtime(temphz200-sysvar.slicebegin^);
X            WriteString("    ");
X            ConvertToString(sysvar.contextswitch^,10,FALSE,temp,done);
X            Copy(SPACES,0,14-Length(temp),pstemp);
X            Concat(temp,pstemp,temp);
X            WriteString(temp);
X            IF sysvar.CronActive^ THEN
X               WriteString("ACTIVE");
X            ELSE
X               WriteString("INACTIVE");
X            END;
X            WriteLn;
X            WriteLn;
X            WriteString(PS3);
X            WriteLn;
X
X            s0:=currentprocess;
X            REPEAT
X               IF s0^.pid#1 THEN
X                  s0:=s0^.next
X               END;
X            UNTIL s0^.pid=1;
X            REPEAT
X             WITH s0^ DO
X             IF stmode THEN
X                IF wsp=NIL THEN dmode:=TRUE ELSE dmode:=FALSE END;
X             ELSE
X                IF wsp#NIL THEN dmode:=TRUE ELSE dmode:=FALSE END;
X             END;
X             IF dmode THEN (* display *)
X               WriteString("  ");
X               Copy(SPACES,0,15-Length(name),pstemp);
X               Concat(name,pstemp,temp);
X               WriteString(temp);
X               ConvertCardinal(pid,2,temp);
X               WriteString(temp);
X               WriteString("  ");
X               ConvertCardinal(ppid,2,temp);
X               WriteString(temp);
X               WriteString("   ");
X               IF active THEN
X                  IF ready THEN
X                     WriteString("Run    ");
```

```
X                ELSE
X                   WriteString("Ready  ");
X                END;
X             ELSE
X                IF wsp=NIL THEN
X                   WriteString("Inactive");
X                ELSE
X                   WriteString("Sleep  ");
X                END;
X             END;
X             ConvertCardinal(pri,2,temp);
X             WriteString(temp);
X             WriteString("  ");
X             FormatTD(time,date);
X             showtime(tick);
X             WriteString("  ");
X             CASE Iport OF
X               printer :       WriteString("P"); |
X               aux     :       WriteString("A"); |
X               con     :       WriteString("C"); |
X               midi    :       WriteString("M"); |
X               null    :       WriteString("N"); |
X               dev0    :       WriteString("0"); |
X               dev1    :       WriteString("1"); |
X               dev2    :       WriteString("2"); |
X               dev3    :       WriteString("3"); |
X               dev4    :       WriteString("4"); |
X               dev5    :       WriteString("5"); |
X               dev6    :       WriteString("6"); |
X               dev7    :       WriteString("7"); |
X             END;
X             WriteString("/");
X             CASE Oport OF
X               printer :       WriteString("P"); |
X               aux     :       WriteString("A"); |
X               con     :       WriteString("C"); |
X               midi    :       WriteString("M"); |
X               null    :       WriteString("N"); |
X               dev0    :       WriteString("0"); |
X               dev1    :       WriteString("1"); |
X               dev2    :       WriteString("2"); |
X               dev3    :       WriteString("3"); |
X               dev4    :       WriteString("4"); |
X               dev5    :       WriteString("5"); |
X               dev6    :       WriteString("6"); |
X               dev7    :       WriteString("7"); |
X             END;
X             WriteLn;
X           END; (* if *)
X           END; (* with *)
X             s0:=s0^.next;
X         UNTIL (s0^.pid=0);
X         WriteLn;
```

```
X           ConvertToString(sysvar.sysmemsize^,10,FALSE,temp,done);
X           WriteString("  System Process Memory Remaining ");
X           WriteString(temp);
X           WriteLn;
X           WriteLn;
X        OldTerm;
XEND     ps.
SHAR_EOF
chmod 0600 PS.MOD || echo "restore of PS.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > REDIR.MOD &&
X
X(*              Copyright 1987 fred brooks LogicTek             *)
X(*                                                              *)
X(*                                                              *)
X(*   First Release                      12/8/87-FGB             *)
X(*                                                              *)
X
XMODULE redir;
X(*$T- *)
XFROM SYSTEM IMPORT ADDRESS, CODE, REGISTER, SETREG;
XFROM GEMX   IMPORT BasePageAddress, BasePageType ;
XFROM GEMDOS IMPORT TermRes;
XFROM XBIOS  IMPORT SuperExec;
XVAR         cmd,dev,a7,sr,port,oldbios  :       ADDRESS;
X            bios [0b4H]                 :       ADDRESS;
X            aline [28H]                 :       ADDRESS;
X            gem   [88H]                 :       ADDRESS;
X            char                        :       CHAR;
X
X(*$P-,$S- *)
XPROCEDURE       changedevbios;
XBEGIN
X        CODE(48e7H,0fffeH);     (* save regs *)
X        CODE(306fH,60);         (* move.w 60(a7),a0  get sr *)
X        sr:=REGISTER(8);
X        IF sr<3ffH THEN        (* called from user mode *)
X           CODE(204fH);            (* move.l a7,a0 *)
X           a7:=REGISTER(8);        (* save ssp *)
X           CODE(4e68H,2e48H);      (* move.l usp,a0  move.l a0,a7 *)
X           CODE(306fH,2);  (* move.w 2(a7),a0 *)
X           dev:=REGISTER(8);
X           CODE(306fH,0);  (* move.w 0(a7),a0 *)
X           cmd:=REGISTER(8);
X           IF (cmd=1) OR (cmd=2) OR (cmd=3) OR (cmd=8) THEN
X              IF dev=2 THEN
X                 dev:=port; (* change to port *)
X                 SETREG(8,dev);
X                 CODE(3f48H,2); (* move.w a0,2(a7) set value in stack *)
X              END;
X           END;
X           SETREG(8,a7);
X           CODE(2e48H);            (* move.l a0,a7 *)
X        ELSE                       (* called from super mode *)
```

```
X           CODE(306fH,68);  (* move.w 68(a7),a0 *)
X           dev:=REGISTER(8);
X           CODE(306fH,66);  (* move.w 66(a7),a0 *)
X           cmd:=REGISTER(8);
X           IF (cmd=1) OR (cmd=2) OR (cmd=3) OR (cmd=8) THEN
X              IF dev=2 THEN
X                 dev:=port; (* change to port *)
X                 SETREG(8,dev);
X                 CODE(3f48H,68); (* move.w a0,68(a7) set value in stack *)
X              END;
X           END;
X        END;
X        SETREG(8,oldbios);  (* move trap adr *)
X        CODE(43faH,10); (* lea 12(pc),a1 *)
X        CODE(2288H); (* move.l a0,(a1) *)
X        CODE(4cdfH,7fffH); (* restore regs movem *)
X        CODE(4ef9H,0,0) (* jmp back to routine *)
XEND             changedevbios;
X(*$P+,$S+ *)
X
X(*$P-,$S- *)
XPROCEDURE       dummyvector;
XBEGIN
X        CODE(4e73H); (* rte *)
XEND             dummyvector;
X(*$P+,$S+ *)
X
X(*$P-,$S- *)
XPROCEDURE       setup;
XBEGIN
X        oldbios:=bios;
X        bios:=ADDRESS(changedevbios);
X(*        aline:=ADDRESS(dummyvector); *)
X        gem:=ADDRESS(dummyvector);
X        CODE(4e75H);    (* RTS *)
XEND             setup;
X(*$P+,$S+ *)
X
XBEGIN;
X    char:=BasePageAddress^.CmdLine[1];
X    IF (char='m') OR (char='M') THEN
X       port:=3;   (* midi *)
X    ELSE
X       port:=1;   (* aux  *)
X    END;
X    SuperExec(setup);
X    WITH BasePageAddress^ DO
X    TermRes(CodeLen+BssLen+LONGCARD(CodeBase-ADDRESS(BasePageAddress)),0);
X    END;
XEND redir.
SHAR_EOF
chmod 0600 REDIR.MOD || echo "restore of REDIR.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > RUN.MOD &&
```

```
X
X(*              Copyright 1988 fred brooks LogicTek            *)
X(*                                                             *)
X(*                                                             *)
X(*   First Release                      3/26/88-FGB            *)
X(*                                                             *)
X
XMODULE run; (* MX2 program pterm code buffer program *)
XFROM    GEMDOS  IMPORT  Term,Exec,ExecMode;
XFROM    GEMX    IMPORT  BasePageAddress;
XFROM    SCANNER IMPORT  scinit,nxparm,ltext,etext,bkparm;
XFROM    SYSTEM  IMPORT  ADR,ADDRESS;
XFROM    Strings IMPORT  String,Length,Pos,Concat;
XVAR     i                       :       INTEGER;
X        p                       :       CARDINAL;
X        ok                      :       BOOLEAN;
X        command,name,tail,temp  :       String;
XBEGIN
X        FOR i:=1 TO 79 DO
X            command[i-1]:=BasePageAddress^.CmdLine[i];
X        END;
X        command[ORD(BasePageAddress^.CmdLine[0])]:=CHAR(0);
X        scinit(ADR(command),SIZE(command));
X        nxparm;
X        ltext(ADR(name),SIZE(name));
X        nxparm;
X        etext(ADR(tail[1]),SIZE(tail));
X        bkparm;
X        etext(ADR(temp),SIZE(temp));
X        tail[0]:=CHAR(Length(temp));
X        IF Pos(name,".",0,p) THEN
X                Exec(loadExecute,name,tail,"",i);
X        ELSE
X                temp:=name;
X                Concat(name,".prg",name);
X                Exec(loadExecute,name,tail,"",i);
X                IF i=(-33) THEN
X                        name:=temp;
X                        Concat(name,".tos",name);
X                        Exec(loadExecute,name,tail,"",i);
X                END;
X                IF i=(-33) THEN
X                        name:=temp;
X                        Concat(name,".ttp",name);
X                        Exec(loadExecute,name,tail,"",i);
X                END;
X        END;
X        ok:=Term(i);
XEND     run.
X
SHAR_EOF
chmod 0600 RUN.MOD || echo "restore of RUN.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > SCANNER.DEF &&
```

```
X
X(*              Copyright 1987 fred brooks LogicTek             *)
X(*                                                              *)
X(*                                                              *)
X(*   First Release                      12/8/87-FGB             *)
X(*                                                              *)
X
XDEFINITION MODULE       SCANNER;
X
XFROM       SYSTEM       IMPORT  ADDRESS;
X
XTYPE
X        modetype        =       (delimiter,register,translation);
X        modes           =       SET OF modetype;
X        scstate         =
X                RECORD
X                        return          :       LONGINT;
X                        auxreturn       :       LONGINT;
X                        disp            :       CARDINAL;
X                        prevdisp        :       CARDINAL;
X                        delim           :       ADDRESS;
X                        delimret        :       CARDINAL;
X                        bufadr          :       ADDRESS;
X                        buflen          :       CARDINAL;
X                        mode            :       modes;
X                END;
X
XVAR     state           :       scstate;
X
XPROCEDURE       scinit(bufadr: ADDRESS; buflen: CARDINAL);
XPROCEDURE       gtdisp(): CARDINAL;
XPROCEDURE       gbdisp(): CARDINAL;
XPROCEDURE       stdisp(disp: CARDINAL);
XPROCEDURE       nxparm;
XPROCEDURE       bkparm;
XPROCEDURE       dlim;
XPROCEDURE       stmode(mode: modes);
XPROCEDURE       char(VAR ch: CHAR);
XPROCEDURE       bkchar(VAR ch: CHAR);
XPROCEDURE       nxchar(VAR ch: CHAR);
XPROCEDURE       onenum;
XPROCEDURE       ltext(bufadr: ADDRESS; buflen: CARDINAL);
XPROCEDURE       etext(bufadr: ADDRESS; buflen: CARDINAL);
XEND                     SCANNER.
SHAR_EOF
chmod 0600 SCANNER.DEF || echo "restore of SCANNER.DEF fails"
sed 's/^X//' << 'SHAR_EOF' > SCANNER.MOD &&
X
X(*              Copyright 1987 fred brooks LogicTek             *)
X(*                                                              *)
X(*                                                              *)
X(*   First Release                      12/8/87-FGB             *)
X(*                                                              *)
```

```
X
XIMPLEMENTATION MODULE  SCANNER;
X(*$T-,$S-,$A+ *)
XFROM    SYSTEM  IMPORT  ADDRESS,ADR;
X
XTYPE
X        charptr         =       POINTER TO CHAR;
X        chset           =       SET OF CHAR;
X(*             internal state record
X        scstate =
X                RECORD
X                        return          :       LONGINT;
X                        auxreturn       :       LONGINT;
X                        disp            :       CARDINAL;
X                        prevdisp        :       CARDINAL;
X                        delim           :       ADDRESS;
X                        delimret        :       CARDINAL;
X                        bufadr          :       ADDRESS;
X                        buflen          :       CARDINAL;
X                        mode            :       modes;
X                END;
X*)
X
XVAR     chrptr                          :       charptr;
X        cmdline,textline                :       ARRAY [0..81] OF CHAR;
X        ch                              :       CHAR;
X
XCONST   alpha   = chset{"A".."Z","a".."z"};
X        num     = chset{"+","-","0".."9"};
X(* Initialize the scanner and pass a buffer to be scanned. Only one buffer
Xmay be processed by the scanner at a time.                          *)
XPROCEDURE       scinit(bufadr: ADDRESS; buflen: CARDINAL);
XBEGIN
X        state.return:=0;
X        state.auxreturn:=0;
X        state.disp:=0;
X        state.prevdisp:=0;
X        state.delim:=NIL;
X        state.bufadr:=bufadr;
X        state.buflen:=buflen;
X        state.mode:=modes{};
X        state.delimret:=0;
XEND             scinit;
X
XPROCEDURE       gtdisp(): CARDINAL;
XBEGIN
XEND             gtdisp;
X
XPROCEDURE       gbdisp(): CARDINAL;
XBEGIN
XEND             gbdisp;
X
X(* set new value as the current scanner displacement *)
```

```
XPROCEDURE       stdisp(disp: CARDINAL);
XBEGIN
X        state.prevdisp:=state.disp;
X        IF bufeol(disp)=(-2) THEN (* past end of line *)
X           state.return:=(-2);
X        ELSE
X           state.prevdisp:=state.disp;
X           state.disp:=disp;
X           state.return:=0;
X        END;
XEND             stdisp;
X
XPROCEDURE       nxparm;
XVAR                i       :       CARDINAL;
X                   ch      :       CHAR;
XBEGIN
X        chrptr:=charptr(state.bufadr+ADDRESS(state.disp));
X        i:=state.disp;
X        LOOP
X                IF bufeol(i)<0 THEN
X                   state.return:=(-2); (* end of line *)
X                   EXIT;
X                END;
X                IF NOT delimit(chrptr^) THEN
X                   state.prevdisp:=state.disp;
X                   state.disp:=i;
X                   state.return:=0;
X                   EXIT;
X                END;
X                INC(i);
X                INC(chrptr);
X        END;
XEND             nxparm;
X
X(* return 0=not eol, -1=eol, -2=past eol *)
XPROCEDURE       bufeol(pos: CARDINAL): INTEGER;
XVAR             cptr        :          charptr;
X                eol         :          BOOLEAN;
X                i           :          CARDINAL;
XBEGIN
X        eol:=FALSE;
X        i:=0;
X        LOOP
X        cptr:=charptr(state.bufadr+ADDRESS(i));
X        IF i=state.buflen THEN
X           EXIT;
X        END;
X        IF cptr^='$' THEN (* search for " $" sequence *)
X           DEC(cptr);
X           IF cptr^=' ' THEN
X              DEC(i);
X              EXIT; (* i = eol *)
X           END;
```

```
X        END;
X        INC(i);
X        END; (* loop *)
X
X        IF (pos=i-1) THEN
X           state.delimret:=2;
X           RETURN (-1);
X        END;
X        IF (pos>i-1) THEN
X           state.delimret:=2;
X           RETURN (-2);
X        END;
X        RETURN 0;
XEND     bufeol;
X
XPROCEDURE       delimit(ch: CHAR): BOOLEAN;
XBEGIN
X        IF (ch=' ') OR (ch=CHAR(0)) THEN
X           state.delimret:=0;
X           RETURN TRUE;
X        END;
X        IF ch=',' THEN
X           state.delimret:=1;
X           RETURN TRUE;
X        END;
X        RETURN FALSE;
XEND             delimit;
X
XPROCEDURE       bkparm;
XBEGIN
X        state.disp:=state.prevdisp;
XEND             bkparm;
X
XPROCEDURE       dlim;
XBEGIN
XEND             dlim;
X
XPROCEDURE       stmode(mode: modes);
XBEGIN
XEND             stmode;
X
XPROCEDURE       char(VAR ch: CHAR);
XBEGIN
X        state.return:=0;
X        chrptr:=charptr(state.bufadr+ADDRESS(state.disp));
X        IF chrptr^#CHAR(0) THEN
X           IF state.disp<state.buflen-1 THEN
X              INC(state.disp);
X           ELSE
X              state.return:=(-2);
X           END;
X        ELSE
X           state.return:=(-2);
```

```
X        END;
X        ch:=chrptr^;
XEND             char;
X
XPROCEDURE       bkchar(VAR ch: CHAR);
XBEGIN
X        state.return:=0;
X        IF state.disp=0 THEN
X           state.return:=(-2);
X           ch:=CHAR(0);
X           RETURN;
X        END;
X        chrptr:=charptr(state.bufadr+ADDRESS(state.disp-1));
X        ch:=chrptr^;
XEND             bkchar;
X
XPROCEDURE       nxchar(VAR ch: CHAR);
XBEGIN
X        state.return:=0;
X        chrptr:=charptr(state.bufadr+ADDRESS(state.disp));
X        IF chrptr^#CHAR(0) THEN
X           IF state.disp<state.buflen-1 THEN
X           ELSE
X              state.return:=(-2);
X           END;
X        ELSE
X           state.return:=(-2);
X        END;
X        ch:=chrptr^;
XEND             nxchar;
X
XPROCEDURE       onenum;
XBEGIN
XEND             onenum;
X
XPROCEDURE       ltext(bufadr: ADDRESS; buflen: CARDINAL);
XVAR                i,j     :       CARDINAL;
X                   textbuf :       charptr;
XBEGIN
X        textbuf:=bufadr;
X        j:=0;
X        chrptr:=charptr(state.bufadr+ADDRESS(state.disp));
X        i:=state.disp;
X        state.prevdisp:=state.disp;
X        IF delimit(chrptr^) THEN
X           textbuf^:=CHAR(0);
X           state.return:=(-1);
X           RETURN;
X        END;
X
X        LOOP
X                IF bufeol(i)=(-2) THEN (* past eol *)
X                   state.return:=(-2);
```

```
X                   EXIT;
X                END;
X                IF NOT delimit(chrptr^) THEN
X                   IF j<buflen THEN (* stop at end of buffer *)
X                      textbuf^:=chrptr^;
X                   END;
X                   state.disp:=i;
X                ELSE
X                   INC(j);
X                   IF NOT (j>buflen) THEN
X                      textbuf^:=CHAR(0);
X                   END;
X                   IF bufeol(i)=(-1) THEN (* if eol *)
SHAR_EOF
echo "End of part 6, continue with part 7"
echo "7" > s2_seq_.tmp
exit 0


From -6833759609631003807
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,c59b2f53012e8a05,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-12-27 13:35:10 PST
Path: utzoo!attcan!uunet!husc6!bloom-beacon!tut.cis.ohio-state.edu!rutgers!ucsd!
ucbvax!agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i087: mx2v230 -- Multitasking kernel with utilities  part07/08
Keywords: shar, Modula-2
Message-ID: <5873@saturn.ucsc.edu>
Date: 27 Dec 88 21:35:10 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1044
Approved: koreth@ssyx.ucsc.edu

Submitted-by: madsen@sask.usask.ca (Jorgen Madsen)
Posting-number: Volume 1, Issue 87
Archive-name: mx2v230/part07

#!/bin/sh
# this is part 7 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file SCANNER.MOD continued
#
CurArch=7
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
```

```
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> SCANNER.MOD
X                       state.return:=(-2);
X                       EXIT;
X                    END;
X                    state.disp:=i;
X                    state.return:=0;
X                    EXIT;
X                 END;
X                 INC(i);
X                 INC(j);
X                 INC(chrptr);
X                 INC(textbuf);
X         END;
XEND              ltext;
X
XPROCEDURE        etext(bufadr: ADDRESS; buflen: CARDINAL);
XVAR                 i,j     :       CARDINAL;
X                    textbuf :       charptr;
XBEGIN
X         textbuf:=bufadr;
X         j:=0;
X         chrptr:=charptr(state.bufadr+ADDRESS(state.disp));
X         i:=state.disp;
X         state.prevdisp:=state.disp;
X         IF delimit(chrptr^) THEN
X            textbuf^:=CHAR(0);
X            state.return:=(-1);
X            RETURN;
X         END;
X
X         LOOP
X                 IF bufeol(i)=(-2) THEN (* past eol *)
X                    state.return:=(-2);
X                    EXIT;
X                 END;
X                 IF chrptr^# 0C THEN
X                    IF j<buflen THEN (* stop at end of buffer *)
X                       textbuf^:=chrptr^;
X                    END;
X                    state.disp:=i;
X                 ELSE
X                    INC(j);
X                    IF NOT (j>buflen) THEN
X                       textbuf^:=CHAR(0);
X                    END;
X                    IF bufeol(i)=(-1) THEN (* if eol *)
X                       state.return:=(-2);
X                       EXIT;
X                    END;
X                    state.disp:=i;
X                    state.return:=0;
```

```
X                   EXIT;
X                END;
X                INC(i);
X                INC(j);
X                INC(chrptr);
X                INC(textbuf);
X        END;
XEND             etext;
X
XBEGIN
XEND                    SCANNER.
SHAR_EOF
chmod 0600 SCANNER.MOD || echo "restore of SCANNER.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > SPAWN.MOD &&
X
X(*              Copyright 1987 fred brooks LogicTek             *)
X(*                                                              *)
X(*                                                              *)
X(*   First Release                      12/8/87-FGB             *)
X(*                                                              *)
X
X(*$A+,$S-,$T- *)
XMODULE  spawn;
XFROM    ATOMIC  IMPORT  MultiBegin,sysvariable,GEMTYPE,ROMDATE,
X                        OTOS,MTOS,OLDDATE,NEWDATE;
XFROM    SYSCALL IMPORT  SysVar,Sleep;
XFROM    SYSTEM  IMPORT  ADDRESS;
XFROM    GEMDOS  IMPORT  OldTerm;
XVAR           sv                              : sysvariable;
X              gem1     [OTOS]                 : POINTER TO GEMTYPE;
X              gem2     [MTOS]                 : POINTER TO GEMTYPE;
X
XBEGIN
X                SysVar(sv);
X                IF ROMDATE=OLDDATE THEN
X                   sv.bpsave^:=gem1^;
X                   sv.bpsave^[0]:=ADDRESS(gem1);
X                END;
X                IF ROMDATE=NEWDATE THEN
X                   sv.bpsave^:=gem2^;
X                   sv.bpsave^[0]:=ADDRESS(gem2);
X                END;
X                MultiBegin;
X                Sleep;
X                OldTerm;
XEND     spawn.
SHAR_EOF
chmod 0600 SPAWN.MOD || echo "restore of SPAWN.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > SPOOLER.MOD &&
X(* this module uses spint 0 to communicate with a controller program
X   Commands are sent via the spintinfo call. and pass values thru
X   the spintcmd variable. Please look at lp.mod for an example.
X   current commands are:
```

```
X   1    examine spool directory. print any files that exist then delete.
X   2    spool queue.
X   99   terminate spooler.prg.
X*)
X
XMODULE spooler;
XFROM   XBIOS    IMPORT  SuperExec;
XFROM   SYSTEM   IMPORT  BYTE,ADR,CODE,ADDRESS;
XFROM   GEMDOS   IMPORT  DirCreate,Open,Close,SetDTA,Delete,SFirst,SNext,
X                        SetPath,Alloc,Free,Read,GetPath,OldTerm;
XFROM   TextIO   IMPORT  WriteString,WriteLn,WriteAdr;
XFROM   BIOS     IMPORT  Device,BCosStat,BConOut;
XFROM   Streams  IMPORT  OpenStream,CloseStream,StreamKinds,Stream;
XFROM   SYSCALL  IMPORT  EnableSpint,SwapProcess,SleepProcess,
X                        WakeupProcess,ProcessPid,DisableSpint;
XTYPE   dtatype  =       RECORD
X                res     :       ARRAY [0..20] OF BYTE;
X                attr    :       BYTE;
X                time    :       CARDINAL;
X                date    :       CARDINAL;
X                size    :       LONGCARD;
X                name    :       ARRAY [0..13] OF CHAR;
X                        END;
XVAR    result,i,return,d,pid                   :       INTEGER;
X       DTA                                     :       dtatype;
X       ok                                      :       BOOLEAN;
X       count                                   :       INTEGER;
X       ch                                      :       BYTE;
X       S                                       :       Stream;
X       endpos,currentpos,delay                 :       LONGCARD;
X       C,char                                  :       POINTER TO CHAR;
X       spintcmd                                :       ARRAY [0..1] OF
X                                                       LONGCARD;
XPROCEDURE       sq;
XBEGIN
X          WriteLn;
X          IF currentpos#endpos THEN
X               WriteString("Printing ");
X               WriteString(DTA.name);
X               WriteAdr(ADDRESS(endpos),7);
X               WriteString(" total characters");
X               WriteAdr(ADDRESS(endpos-currentpos),7);
X               WriteString(" characters left to print.");
X          ELSE
X               WriteString("Nothing.");
X          END;
X          WriteLn;
XEND             sq;
X
XPROCEDURE       run;
XBEGIN
X        IF (spintcmd[0]=1) OR (spintcmd[0]=99) THEN
X          WakeupProcess(pid);
```

```
X        END;
X        IF spintcmd[0]=2 THEN
X           sq;
X        END;
XEND             run;
X
XBEGIN
X        spintcmd[1]:=LONGCARD(sq);
X        pid:=ProcessPid();
X        ok:=SetPath("\");
X        ok:=DirCreate("mx2spool");
X        IF ok THEN
X           WriteString("Creating spool directory -> \MX2SPOOL");
X           WriteLn;
X        END;
X        ok:=SetPath("\mx2spool");
X        IF NOT ok THEN
X           DisableSpint(0);
X           WriteString("Unable to create or use MX2SPOOL directory");
X           WriteLn;
X           OldTerm;
X        END;
X        ok:=EnableSpint(0,run,ADR(spintcmd)); (* spint to start spooler *)
X        SetDTA(ADR(DTA));
X        LOOP
X           SFirst("*.*",0,result);
X           WHILE result=0 DO
X
X        count:=0;
X        OpenStream(S,DTA.name,READ,return);
X        IF return<0 THEN
X        ELSE
X           currentpos:=0;
X           endpos:=S.endPos;
X           Alloc(endpos,C);
X           char:=C;
X           IF char#NIL THEN
X              Read(S.handle,endpos,char);
X              CloseStream(S,return);
X              WHILE (currentpos#endpos) DO
X                 WHILE NOT BCosStat(PRT) DO
X                    INC(delay);
X                    IF delay>1000 THEN
X                       delay:=0;
X                       SwapProcess;
X                    END;
X                 END;
X                 BConOut(PRT,char^);
X                 INC(currentpos);
X                 INC(char);
X                 INC(count);
X                 IF count>32 THEN
X                    count:=0;
```

```
X                    SwapProcess;
X                    IF spintcmd[0]=99 THEN
X                       DisableSpint(0);
X                       OldTerm;
X                    END;
X                 END;
X              END;
X              IF Free(C) THEN END;
X              BConOut(PRT,CHAR(0cH)); (* send formfeed *)
X           ELSE
X              CloseStream(S,return);
X              WriteString(DTA.name);
X              WriteString(" * SPOOLER OUT OF MEMORY *");
X              WriteLn;
X           END;
X        END;
X           ok:=Delete(DTA.name);
X           SNext(result);
X           END;
X           SleepProcess(pid);
X           IF spintcmd[0]=99 THEN
X              DisableSpint(0);
X              OldTerm;
X           END;
X        END;
XEND    spooler.
SHAR_EOF
chmod 0600 SPOOLER.MOD || echo "restore of SPOOLER.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > SQ.MOD &&
XMODULE  sq;
XFROM    SYSCALL         IMPORT  SpintInfo;
XFROM    GEMDOS          IMPORT  Term;
XFROM    Terminal        IMPORT  WriteString,WriteLn;
XVAR     spintcmd        :       POINTER TO ARRAY [0..1] OF LONGCARD;
X        SQ              :       PROC;
X        ok              :       BOOLEAN;
XBEGIN
X        IF SpintInfo(0,spintcmd) THEN
X           SQ:=PROC(spintcmd^[1]);
X           SQ();
X        ELSE
X           WriteLn;
X           WriteString("Background spooler not running");
X           WriteLn;
X           ok:=Term(-1);
X        END;
X        ok:=Term(0);
XEND     sq.
SHAR_EOF
chmod 0600 SQ.MOD || echo "restore of SQ.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > SUBMIT.MOD &&
X
X(*              Copyright 1987 fred brooks LogicTek             *)
```

```
X(*                                                                  *)
X(*                                                                  *)
X(*   First Release                      12/8/87-FGB                 *)
X(*                                                                  *)
X
XMODULE submit;
X
XFROM SYSTEM IMPORT ADDRESS;
XFROM GEMX   IMPORT BasePageAddress ;
XFROM XBIOS  IMPORT IOREC,IORECPTR,SerialDevice,IORec;
XFROM    GEMDOS  IMPORT  OldTerm;
XVAR         char                       :       CHAR;
X            i                          :       CARDINAL;
X            kbdiorec                   :       IORECPTR;
X            ibuf                       :       POINTER TO ARRAY [0..63] OF
X                                               LONGCARD;
X
XBEGIN;
X    kbdiorec:=IORec(Keyboard);
X    ibuf:=kbdiorec^.ibuf;
X    char:=BasePageAddress^.CmdLine[1];
X    FOR i:=1 TO ORD(BasePageAddress^.CmdLine[0])+1 DO
X        ibuf^[i]:=LONGCARD(BasePageAddress^.CmdLine[i]);
X    END;
X    ibuf^[ORD(BasePageAddress^.CmdLine[0])+1]:=LONGCARD(0dH);
X    kbdiorec^.ibufhd:=0;
X    kbdiorec^.ibuftl:=(ORD(BasePageAddress^.CmdLine[0])+1)*4;
X    OldTerm;
XEND submit.
SHAR_EOF
chmod 0600 SUBMIT.MOD || echo "restore of SUBMIT.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > SUBMITA.MOD &&
X
X(*              Copyright 1987 fred brooks LogicTek                 *)
X(*                                                                  *)
X(*                                                                  *)
X(*   First Release                      12/8/87-FGB                 *)
X(*                                                                  *)
X
X(*$A+ *)
XMODULE submita;
XFROM SYSTEM IMPORT ADDRESS;
XFROM GEMX   IMPORT BasePageAddress ;
XFROM XBIOS  IMPORT IOREC,IORec,IORECPTR,SerialDevice;
XFROM GEMDOS IMPORT OldTerm;
XVAR         char                       :      CHAR;
X            i,count                    :      CARDINAL;
X            kbdiorec                   :      IORECPTR;
X            ibuf                       :      POINTER TO ARRAY [0..255] OF
X                                               CHAR;
X
XBEGIN;
X    kbdiorec:=IORec(RS232);
```

```
X    ibuf:=kbdiorec^.ibuf;
X    char:=BasePageAddress^.CmdLine[1];
X    count:=ORD(BasePageAddress^.CmdLine[0])+1;
X    IF count>78 THEN count:=78 END;
X    FOR i:=1 TO count DO
X        ibuf^[i]:=CHAR(BasePageAddress^.CmdLine[i]);
X    END;
X    ibuf^[count]:=CHAR(0dH);
X    kbdiorec^.ibufhd:=0;
X    kbdiorec^.ibuftl:=count;
X    OldTerm;
XEND submita.
SHAR_EOF
chmod 0600 SUBMITA.MOD || echo "restore of SUBMITA.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > SUBMITM.MOD &&
X
X(*              Copyright 1987 fred brooks LogicTek            *)
X(*                                                             *)
X(*                                                             *)
X(*   First Release                      12/8/87-FGB            *)
X(*                                                             *)
X
X(*$A+ *)
XMODULE submita;
XFROM SYSTEM IMPORT ADDRESS;
XFROM GEMX   IMPORT BasePageAddress ;
XFROM XBIOS  IMPORT IOREC,IORec,IORECPTR,SerialDevice;
XFROM GEMDOS IMPORT OldTerm;
XVAR         char                        :      CHAR;
X            i,count                     :      CARDINAL;
X            kbdiorec                    :      IORECPTR;
X            ibuf                        :      POINTER TO ARRAY [0..127] OF
X                                               CHAR;
X
XBEGIN;
X    kbdiorec:=IORec(MIDI);
X    ibuf:=kbdiorec^.ibuf;
X    char:=BasePageAddress^.CmdLine[1];
X    count:=ORD(BasePageAddress^.CmdLine[0])+1;
X    IF count>78 THEN count:=78 END;
X    FOR i:=2 TO count DO
X        ibuf^[i]:=CHAR(BasePageAddress^.CmdLine[i-1]);
X    END;
X    ibuf^[count+1]:=CHAR(0dH);
X    kbdiorec^.ibufhd:=0;
X    kbdiorec^.ibuftl:=count+1;
X    OldTerm;
XEND submita.
SHAR_EOF
chmod 0600 SUBMITM.MOD || echo "restore of SUBMITM.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > SYS.MOD &&
X
X(*              Copyright 1987 fred brooks LogicTek            *)
```

```
X(*                                                                  *)
X(*                                                                  *)
X(*   First Release                      12/8/87-FGB                 *)
X(*                                                                  *)
X
X(*$S-,$T- *)
XMODULE  SYS;
XFROM    SYSCALL IMPORT  SysCmd;
XFROM    GEMDOS  IMPORT  OldTerm;
XBEGIN
X        SysCmd; (* submit to kernal *)
X        OldTerm;
XEND     SYS.
SHAR_EOF
chmod 0600 SYS.MOD || echo "restore of SYS.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > SYSCALL.DEF &&
X
X(*              Copyright 1987 fred brooks LogicTek                 *)
X(*                                                                  *)
X(*                                                                  *)
X(*   First Release                      12/8/87-FGB                 *)
X(* Add ProcessPid to return PID of calling process                  *)
X(*                                     12/12/87-FGB                 *)
X(* Added variable parm to StartProcess to pass info to process      *)
X(* in currentprocess.gemsave[15]       1/1/88-FGB                   *)
X(*                                                                  *)
X
XDEFINITION MODULE SYSCALL;
X
XFROM    ATOMIC  IMPORT  SIGNAL,sysvariable;
XFROM    SYSTEM  IMPORT  ADDRESS;
XFROM    Strings IMPORT  String;
X
XCONST   (* SYSTEM ERRORS *)
X
X        ENOERR           =       0;     (* no error *)
X        ENOENT           =       2;     (* no file or directory *)
X        ENORCH           =       3;     (* no such process *)
X        ENOMEM           =       12;    (* out of memory *)
X        EBADRQC          =       54;    (* bad request *)
X
X(* Read the program command line and pass it to kernel to execute *)
XPROCEDURE   SysCmd;
X
X(* Pass command line to kernal to execute *)
XPROCEDURE   SysReq(VAR command: ARRAY OF CHAR);
X
X(* Get the processdescriptor for the currentprocess *)
XPROCEDURE  SysDes(VAR currentprocess : SIGNAL);
X
X(* Get the systemvariables ,These are copies of the variables *)
XPROCEDURE  SysVar(VAR sysvar : sysvariable);
X
```

```
X(* Return PID of calling process *)
XPROCEDURE  ProcessPid(): INTEGER;
X
X(* Put procedure to sleep *)
XPROCEDURE        Sleep;
X
X(* check to see if special interrupt has happened , execute if triggered *)
XPROCEDURE CheckSpint;
X
X(*  Setup spint linkage to spintid *)
XPROCEDURE EnableSpint(spintid: CARDINAL; routine: PROC; data: ADDRESS): BOOLEAN;
X
X(*  request spint data and active info  return TRUE if active bit set *)
X(*  data will be set to NIL if the spint is disabled           *)
XPROCEDURE SpintInfo(spintid: CARDINAL; VAR data: ADDRESS): BOOLEAN;
X
X(*  remove spint linkage to spintid *)
XPROCEDURE DisableSpint(spintid: CARDINAL);
X
X(*  send spint to routine  *)
XPROCEDURE Trigger(spintid: CARDINAL): BOOLEAN;
X
X(*  Hold program interrupts *)
XPROCEDURE HoldSpint(spintid: CARDINAL);
X
X(*  Release program interrupts *)
XPROCEDURE ReleaseSpint(spintid: CARDINAL);
X
X(*  Wait a specified period of time for a program interrupt ticks are
X    in 200 hz clocks. If tick is -1 then wait forever for interrupt  *)
XPROCEDURE IntDelay(tick: LONGINT): INTEGER;
X
X(* create a process for the MX2 system, place in the scheduler ready
X   list, start new process *)
XPROCEDURE StartProcess(VAR P: PROC;
X                       VAR n: LONGCARD;
X                       VAR priority: INTEGER;
X                       VAR pn: String;
X                       VAR parm: ADDRESS);
X
X(* store the currentprocess and switch to next process in ready list *)
XPROCEDURE SwapProcess;
X
X(* end process and remove it from the ready list, free memory used by
X   process *)
XPROCEDURE TermProcess(VAR id: INTEGER);
X
X(* return the next pid the system will use for a new process *)
XPROCEDURE NextPid(): INTEGER;
X
X(* tell scheduler not to run this process but keep in memory *)
XPROCEDURE SleepProcess(VAR id: INTEGER);
X
```

```
X(* tell scheduler tp sleep for msec 1000th of a second  *)
XPROCEDURE DozeProcess(VAR id: INTEGER; VAR msec: LONGCARD);
X
X(* Sleep process until the contents of 'loc' equals 'value AND mask'.
Xmsec is the timeout value in milliseconds if set the 0 it waits forever *)
XPROCEDURE WaitProcess(VAR id: INTEGER; VAR loc: ADDRESS;
X                      VAR value,mask,msec: LONGCARD);
X
X(* tell scheduler to start running this process again if it was sleeping
X   before *)
XPROCEDURE WakeupProcess(VAR id: INTEGER);
X
X(* change process priority *)
XPROCEDURE ChangeProcessPriority(VAR id: INTEGER; VAR pri: INTEGER);
X
X(* turn on the scheduler interrupt, and start normal process switching *)
XPROCEDURE MultiBegin;
X
X(* turn off the scheduler interrupt, used to stop process switching in
X   section of code that should not be swapped out *)
XPROCEDURE MultiEnd;
X
XEND               SYSCALL.
SHAR_EOF
chmod 0600 SYSCALL.DEF || echo "restore of SYSCALL.DEF fails"
sed 's/^X//' << 'SHAR_EOF' > SYSCALL.MOD &&
X
X(*              Copyright 1987 fred brooks LogicTek             *)
X(*                                                              *)
X(*                                                              *)
X(*   First Release                      12/8/87-FGB             *)
X(* Added variable parm to StartProcess to pass info to process  *)
X(* in currentprocess.gemsave[15]        1/1/88-FGB              *)
X(* Fixed SpintInfo to give info anytime. It was giving info     *)
X(* only after a spint was enabled before it was executed.       *)
X(*                                      5/21/88-FGB             *)
X(*                                                              *)
X
X(*$S-,$T- *)
XIMPLEMENTATION MODULE  SYSCALL;
XFROM    ATOMIC  IMPORT  sysvariable,CheckFlag,SetFlag,MAGIC,SIGNAL,
X                        spinttype;
XFROM    XBIOS   IMPORT  SuperExec;
XFROM    SYSTEM  IMPORT  CODE,ADDRESS;
XFROM    GEMX    IMPORT  BasePageAddress;
XFROM    GEMDOS  IMPORT  GetPath,GetDrv,Super;
XFROM    Strings IMPORT  String;
X
XVAR     sysvar                  :       sysvariable;
X        sysvector [144H]        :       POINTER TO sysvariable;
X        goodbye                 :       BOOLEAN;
X        i,wait,drv              :       CARDINAL;
X        currentprocess          :       SIGNAL;
```

```
X        ssv                     :       ADDRESS;
X
X(*$P- *)
XPROCEDURE       submit;
XBEGIN
X        sysvar:=sysvector^;
X        CODE(4e75H); (* rts *)
XEND     submit;
X(*$P+ *)
X
XPROCEDURE       SysCmd;
XBEGIN
X        MultiEnd;
X        SuperExec(submit);
X
X        FOR i:=1 TO 79 DO
X            sysvar.command^[i-1]:=BasePageAddress^.CmdLine[i];
X        END;
X        sysvar.command^[ORD(BasePageAddress^.CmdLine[0])]:=CHAR(0);
X
X        MultiBegin;
X        (* wait until no commands are being executed *)
X        REPEAT
X        UNTIL NOT CheckFlag(sysvar.request^.req);
X        MultiEnd;
X
X        (* tell SCHED that we want him to show his face *)
X        sysvar.request^.magic:=MAGIC;
X        currentprocess:=sysvar.currentprocess^;
X        sysvar.request^.pid:=currentprocess^.pid;
X        GetPath(currentprocess^.ipenvstr,0);
X        GetDrv(drv);
X        currentprocess^.flags[0]:=LONGCARD(drv);
X        SetFlag(sysvar.request^.req);
X        MultiBegin;
XEND     SysCmd;
X
XPROCEDURE       SysReq(VAR command: ARRAY OF CHAR);
XBEGIN
X        MultiEnd;
X        SuperExec(submit);
X
X        FOR i:=0 TO HIGH(command)-1 DO
X            sysvar.command^[i]:=command[i];
X        END;
X        sysvar.command^[HIGH(command)]:=0C;
X
X        MultiBegin;
X        (* wait until no commands are being executed *)
X        REPEAT
X        UNTIL NOT CheckFlag(sysvar.request^.req);
X
X        (* tell SCHED that we want him to show his face *)
```

```
X        sysvar.request^.magic:=MAGIC;
X        currentprocess:=sysvar.currentprocess^;
X        sysvar.request^.pid:=currentprocess^.pid;
X        GetPath(currentprocess^.ipenvstr,0);
X        GetDrv(drv);
X        currentprocess^.flags[0]:=LONGCARD(drv);
X        SetFlag(sysvar.request^.req);
X        MultiBegin;
XEND     SysReq;
X
XPROCEDURE       SysDes(VAR cp: SIGNAL);
XBEGIN
X        SuperExec(submit);
X        cp:=sysvar.currentprocess^;
XEND     SysDes;
X
XPROCEDURE       SysVar(VAR sv: sysvariable);
XBEGIN
X        SuperExec(submit);
X        sv:=sysvar;
XEND     SysVar;
X
XPROCEDURE       ProcessPid(): INTEGER;
XBEGIN
X        SuperExec(submit);
X        SysVar(sysvar);
X        currentprocess:=sysvar.currentprocess^;
X        RETURN currentprocess^.pid;
XEND             ProcessPid;
X
XPROCEDURE       Sleep;
XVAR             pid      :       INTEGER;
XBEGIN
X        pid:=ProcessPid();
X        SleepProcess(pid);
XEND     Sleep;
X
XPROCEDURE       FindProcess(pid: INTEGER; cp: SIGNAL): SIGNAL;
XVAR             s                :         SIGNAL;
XBEGIN
X        s:=cp;
X        LOOP                    (* find  process id *)
X          s:=s^.next;
X          IF s^.pid=pid THEN     (* found id *)
X             RETURN s;
X          END;
X          IF s^.pid=cp^.pid THEN      (* id not found in list *)
X             RETURN NIL;
X          END;
X        END;
XEND     FindProcess;
X
XPROCEDURE       CheckSpint;
```

```
XBEGIN
X        SuperExec(submit);
X
X        FOR i:=0 TO 15 DO (* check all spints and run if set *)
X            IF (i IN sysvar.spintenable^)
X               AND (i IN sysvar.spintmask^)
X               AND (ADDRESS(sysvar.spint^[i].proc)#NIL) THEN
X               sysvar.spint^[i].proc;
X            END;
X            EXCL(sysvar.spintenable^,i); (* clear flag after complete *)
X        END;
XEND     CheckSpint;
X
XPROCEDURE       EnableSpint(spintid: CARDINAL; routine: PROC;
X                            data: ADDRESS): BOOLEAN;
XBEGIN
X        SuperExec(submit);
X
X        IF spintid>15 THEN RETURN FALSE END;
X        sysvar.spint^[spintid].proc:=routine;
X        sysvar.spint^[spintid].data:=data;
X        INCL(sysvar.spintmask^,spintid);
X        RETURN TRUE;
XEND     EnableSpint;
X
XPROCEDURE       SpintInfo(spintid: CARDINAL; VAR data: ADDRESS): BOOLEAN;
XBEGIN
X        SuperExec(submit);
X        IF spintid>15 THEN RETURN FALSE END;
X        data:=sysvar.spint^[spintid].data;
X        IF ADDRESS(sysvar.spint^[spintid].proc)#NIL THEN
X           RETURN TRUE;
X        ELSE;
X           data:=NIL;
X           RETURN FALSE;
X        END;
XEND     SpintInfo;
X
XPROCEDURE       DisableSpint(spintid: CARDINAL);
XBEGIN
X        SuperExec(submit);
X
X        IF spintid>15 THEN RETURN END;
X        sysvar.spint^[spintid].proc:=PROC(NIL);
X        EXCL(sysvar.spintenable^,spintid);
X        EXCL(sysvar.spintmask^,spintid);
XEND     DisableSpint;
X
XPROCEDURE       HoldSpint(spintid: CARDINAL);
XBEGIN
X        SuperExec(submit);
X
X        IF spintid>15 THEN RETURN END;
```

```
X        EXCL(sysvar.spintmask^,spintid);
XEND     HoldSpint;
X
XPROCEDURE       ReleaseSpint(spintid: CARDINAL);
XBEGIN
X        SuperExec(submit);
X
X        IF spintid>15 THEN RETURN END;
X        INCL(sysvar.spintmask^,spintid);
XEND     ReleaseSpint;
X
XPROCEDURE       IntDelay(tick: LONGINT): INTEGER;
XVAR             cp      :       SIGNAL;
XBEGIN
X        SysDes(cp);
X        cp^.active:=FALSE; (* put process to sleep *)
X        MultiBegin;
X        REPEAT
X        UNTIL cp^.active; (* wait until active *)
XEND     IntDelay;
X
XPROCEDURE       Trigger(spintid: CARDINAL): BOOLEAN;
XBEGIN
X        SuperExec(submit);
X
X        IF spintid>15 THEN RETURN FALSE END;
X        INCL(sysvar.spintenable^,spintid);
X        RETURN TRUE;
XEND     Trigger;
X
XPROCEDURE StartProcess(VAR P: PROC;
X                       VAR n: LONGCARD;
X                       VAR priority: INTEGER;
X                       VAR pn: String;
X                       VAR parm: ADDRESS);
XBEGIN
X        SuperExec(submit);
X        currentprocess:=sysvar.currentprocess^;
X        sysvar.request^.pid:=currentprocess^.pid;
X        sysvar.StartProcess(P,n,priority,pn,parm);
XEND     StartProcess;
X
XPROCEDURE SwapProcess;
XBEGIN
X        SuperExec(submit);
X        sysvar.SwapProcess;
XEND     SwapProcess;
X
XPROCEDURE TermProcess(VAR id: INTEGER);
XBEGIN
X        SuperExec(submit);
X        sysvar.TermProcess(id);
XEND     TermProcess;
```

```
X
XPROCEDURE NextPid(): INTEGER;
XVAR     Pid     :       INTEGER;
XBEGIN
X        SuperExec(submit);
X        Pid:=sysvar.NextPid();
X        RETURN Pid;
XEND     NextPid;
X
XPROCEDURE SleepProcess(VAR id: INTEGER);
XBEGIN
X        SuperExec(submit);
X        sysvar.SleepProcess(id);
XEND     SleepProcess;
X
XPROCEDURE DozeProcess(VAR id: INTEGER; VAR msec: LONGCARD);
XBEGIN
X        SuperExec(submit);
X        sysvar.DozeProcess(id,msec);
XEND     DozeProcess;
X
XPROCEDURE WaitProcess(VAR id: INTEGER; VAR loc: ADDRESS;
X                      VAR value,mask,msec: LONGCARD);
XBEGIN
X        SuperExec(submit);
X        sysvar.WaitProcess(id,loc,value,mask,msec);
XEND     WaitProcess;
X
XPROCEDURE WakeupProcess(VAR id: INTEGER);
XBEGIN
X        SuperExec(submit);
X        sysvar.WakeupProcess(id);
XEND     WakeupProcess;
X
XPROCEDURE ChangeProcessPriority(VAR id: INTEGER; VAR pri: INTEGER);
XBEGIN
X        SuperExec(submit);
X        sysvar.ChangeProcessPriority(id,pri);
XEND     ChangeProcessPriority;
X
XPROCEDURE MultiBegin;
XBEGIN
X        SuperExec(submit);
X        sysvar.MultiBegin;
XEND     MultiBegin;
X
XPROCEDURE MultiEnd;
XBEGIN
X        SuperExec(submit);
X        sysvar.MultiEnd;
XEND     MultiEnd;
X
XBEGIN
```

```
XEND     SYSCALL.
X
SHAR_EOF
chmod 0600 SYSCALL.MOD || echo "restore of SYSCALL.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > XA.MOD &&
XMODULE  xa;
XFROM    SYSCALL         IMPORT  SpintInfo,Trigger;
XFROM    GEMDOS          IMPORT  Term;
XFROM    Terminal        IMPORT  WriteString,WriteLn;
XVAR     spintcmd        :       POINTER TO ARRAY [0..2] OF LONGCARD;
X        ok              :       BOOLEAN;
XBEGIN
X        IF SpintInfo(1,spintcmd) THEN
X           spintcmd^[0]:=99;
X           ok:=Trigger(1);
X        ELSE
X           WriteLn;
X           WriteString("Background xmodem not running");
X           WriteLn;
X           ok:=Term(-1);
X        END;
X        ok:=Term(0);
XEND     xa.
SHAR_EOF
chmod 0600 XA.MOD || echo "restore of XA.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > XMODEM.DEF &&
X
X(*              Copyright 1987 fred brooks LogicTek             *)
X(*                                                              *)
X(*                                                              *)
X(*   First Release                      12/8/87-FGB             *)
X(*                                                              *)
X
XDEFINITION MODULE XMODEM;
XFROM       Strings  IMPORT    String;
XVAR     xmodemerror     :       INTEGER; (* xmodem errors *)
X        xmodemabort     :       BOOLEAN; (* timeout or abort flag *)
X        crcmode         :       BOOLEAN; (* set crc or checksum *)
X        xfrname         :       String;
X        mdmPacketsSent          :       INTEGER;
X        mdmPacketsReceived      :       INTEGER;
X        mdmBadPackets           :       INTEGER;
X        mdmNakedPackets         :       INTEGER;
X        mdmBytesXferred         :       LONGCARD;
X        endblk                  :       INTEGER;
X
XPROCEDURE       xmodemrec(filename: ARRAY OF CHAR): BOOLEAN;
XPROCEDURE       xmodemsnd(filename: ARRAY OF CHAR): BOOLEAN;
XPROCEDURE       xmodemstat;
X
XEND  XMODEM.
SHAR_EOF
chmod 0600 XMODEM.DEF || echo "restore of XMODEM.DEF fails"
```

```
sed 's/^X//' << 'SHAR_EOF' > XMODEM.MOD &&
X
X(*              Copyright 1987 fred brooks LogicTek             *)
X(*                                                              *)
X(*                                                              *)
X(*   First Release                    12/8/87-FGB              *)
X(*   Minor fixups                     3/7/88-FGB               *)
X(*                                                              *)
X
X(*$T-,$S-,$A+ *)
X(* This version of xmodem has been written using UNIX and the sealink
X   C programming versions as examples. Many thanks to those who have done
X   this before me.        Fred Brooks                                  *)
X
XIMPLEMENTATION MODULE XMODEM;
XFROM SYSTEM     IMPORT ADDRESS, CODE, REGISTER, SETREG, ADR, WORD;
XFROM GEMX       IMPORT BasePageAddress, BasePageType ;
XFROM BIOS       IMPORT BConStat, BCosStat, BConIn, BConOut, Device;
XFROM XBIOS      IMPORT SuperExec;
XFROM GEMDOS     IMPORT Create, Open, Close, Write, Read, GetDTA, SFirst;
XFROM TextIO     IMPORT WriteString, WriteLn, WriteInt,  WriteAdr;
XFROM BitStuff   IMPORT WAnd, WEor, WShl, WShr;
XFROM Strings    IMPORT String, Assign;
X
XTYPE            CharPtr       =          POINTER TO ARRAY [0..MAX(LONGINT)] OF
CHAR;
X
XCONST           SECSIZ        =          80H;
X                BUFSIZ        =          200H;
X                ERRORMAX      =          20;
X                RETRYMAX      =          20;
X                SOH           =          1c;
X                EOT           =          4c;
X                ACK           =          6c;
X                NAK           =          25c;
X                C             =          103c;
X                RTS           =          4e75H;
X                BELL          =          7c;
X                CTRLZ         =          32c;
X
XVAR             result,mtimeout         :       INTEGER;
X                filename                :       String;
X                hz200 [04baH]           :       LONGCARD;
X                t1,prtime               :       LONGCARD;
X                readchar                :       CHAR;
X                filesize                :       POINTER TO LONGCARD;
X                snd,rec,ok              :       BOOLEAN;
X
X(*$P- *)
XPROCEDURE       rdtime();       (* read 200hz clock *)
XBEGIN
X        prtime:=hz200;
X        CODE(RTS);
```

```
XEND             rdtime;
X(*$P+ *)
X
XPROCEDURE       GetTime(): LONGCARD;
XBEGIN
X        SuperExec(rdtime);
X        RETURN prtime;
XEND             GetTime;
X
XPROCEDURE       timerset(time: INTEGER): LONGCARD;
XBEGIN
X        RETURN (LONGCARD(time)+(GetTime() DIV 20));
XEND             timerset;
X
XPROCEDURE       timeup(timer: LONGCARD): BOOLEAN;
XBEGIN
X        IF ((GetTime() DIV 20)>timer) OR ((GetTime() DIV 20)=timer) THEN
X           RETURN TRUE;
X        ELSE
X           RETURN FALSE;
X        END;
XEND             timeup;
X
XPROCEDURE       errorbells;
XVAR             i,delay        :       CARDINAL;
XBEGIN
X           FOR i:=0 TO 3 DO
X               FOR delay:=0 TO 10000 DO END;
X               BConOut(CON,BELL);
X           END;
XEND             errorbells;
X
XPROCEDURE       crcupdate(crcvalue: CARDINAL; data: CHAR): CARDINAL;
XCONST           GEN1X5X12X16                  =     1021H;
XVAR             i,xin,cha                     :     INTEGER;
X                t                             :     CARDINAL;
XBEGIN
X        cha:=INTEGER(data);
X        FOR i:=0 TO 7 DO
X            xin:=INTEGER(WAnd(crcvalue,8000H));
X            cha:=INTEGER(WShl(cha,1));
X            IF INTEGER(WAnd(cha,100H))#0 THEN
X               t:=crcvalue;
X               crcvalue:=1+CARDINAL(WShl(t,1));
X            ELSE
X               t:=crcvalue;
X               crcvalue:=0+CARDINAL(WShl(t,1));
X            END;
X            IF xin#0 THEN
X               crcvalue:=CARDINAL(WEor(crcvalue,GEN1X5X12X16));
X            END;
X        END;
X        RETURN crcvalue;
```

```
XEND             crcupdate;
X
XPROCEDURE       crcfinish(crcvalue: CARDINAL): CARDINAL;
XBEGIN
X        RETURN CARDINAL(WAnd(crcupdate(crcupdate(crcvalue,0c),0c),0ffffH));
X
X
XEND             crcfinish;
X
XPROCEDURE       IAnd255(num: INTEGER): INTEGER;
XBEGIN
X        RETURN INTEGER(WAnd(num,0ffH));
XEND             IAnd255;
X
XPROCEDURE       mdmini;
XBEGIN
X        ok:=FALSE;
X        xmodemerror:=0;
X        xmodemabort:=FALSE;
X        mtimeout:=120;
X        mdmBytesXferred:=0;
X        mdmPacketsSent:=0;
X        mdmPacketsReceived:=0;
X        mdmBadPackets:=0;
X        mdmNakedPackets:=0;
XEND             mdmini;
X
XPROCEDURE       xmodemstat;
XBEGIN
X        WriteLn;
X        WriteString("      XMODEM STATUS       ");
X        IF rec THEN
X           WriteString(" receiver active       ");
X           WriteString(xfrname);
X           IF crcmode THEN
X              WriteString(" CRC mode.");
X           ELSE
X              WriteString(" CHECKSUM mode.");
X           END;
X        END;
X        IF snd THEN
X           WriteString(" transmitter active    ");
X           WriteString(xfrname);
X           IF crcmode THEN
X              WriteString(" CRC mode.");
X           ELSE
X              WriteString(" CHECKSUM mode.");
X           END;
X        END;
X        WriteLn;
X        IF ok THEN
X           WriteString("       Transfer complete.  ");
X           WriteLn;
```

```
X        END;
X        IF xmodemerror#0 THEN
X           WriteString("       Transfer aborted!  ");
X           errorbells;
X           WriteLn;
X        END;
X        WriteLn;
X        WriteString(" Total packets sent  ");
X        WriteInt(mdmPacketsSent,12);
X        WriteLn;
X        WriteString(" Packets left        ");
X        WriteInt(endblk,12);
X        WriteLn;
X        WriteString(" Packets received    ");
X        WriteInt(mdmPacketsReceived,12);
X        WriteLn;
X        WriteString(" Bad packets         ");
SHAR_EOF
echo "End of part 7, continue with part 8"
echo "8" > s2_seq_.tmp
exit 0


From 1847199745852624067
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,b648939ef077609,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-12-27 13:35:14 PST
Path: utzoo!attcan!uunet!husc6!bloom-beacon!tut.cis.ohio-state.edu!rutgers!ucsd!
ucbvax!agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i088: mx2v230 -- Multitasking kernel with utilities  part08/08
Keywords: shar, Modula-2
Message-ID: <5874@saturn.ucsc.edu>
Date: 27 Dec 88 21:35:14 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 590
Approved: koreth@ssyx.ucsc.edu

Submitted-by: madsen@sask.usask.ca (Jorgen Madsen)
Posting-number: Volume 1, Issue 88
Archive-name: mx2v230/part08

#!/bin/sh
# this is part 8 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file XMODEM.MOD continued
#
CurArch=8
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
```

```
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> XMODEM.MOD
X        WriteInt(mdmBadPackets,12);
X        WriteLn;
X        WriteString(" Naked packets sent  ");
X        WriteInt(mdmNakedPackets,12);
X        WriteLn;
X        WriteString(" Bytes transferred   ");
X        WriteAdr(ADDRESS(mdmBytesXferred),12);
X        WriteLn;
XEND             xmodemstat;
X
XPROCEDURE       setbuffer(char: CharPtr; length: CARDINAL; value: CHAR);
XVAR             data            :        POINTER TO CHAR;
XBEGIN
X        WHILE length#0 DO
X              data:=ADDRESS(char);
X              data^:=value;
X              INC(char);
X              DEC(length);
X        END;
XEND             setbuffer;
X
XPROCEDURE       writeModem(char: CharPtr; count: LONGCARD);
XVAR             data             :        POINTER TO CHAR;
XBEGIN
X        WHILE count#0 DO
X              DEC(count);
X              data:=ADDRESS(char);
X              INC(char);
X
X              sendchar(data^);
X        END;
XEND             writeModem;
X
XPROCEDURE       readModem(VAR char: CHAR; time: INTEGER);
XVAR             data         :  CHAR;
X                longchar     :  LONGCARD;
X                t            :  BITSET;
X                WaitTime     :  LONGCARD;
X                ticks        :  CARDINAL;
XBEGIN
X        IF time=0 THEN
X           IF BConStat(AUX) THEN (* return char *)
X              longchar:=BConIn(AUX);
X              t:=BITSET(longchar);
X              EXCL(t,8);
X              char:=CHAR(t);
```

```
X              RETURN;
X           ELSE
X              char:=CHAR(255);
X              RETURN;
X           END;
X        END;
X
X        WaitTime:=LONGCARD(time)+(GetTime() DIV 20);
X        ticks:=0;
X        LOOP
X                IF BConStat(AUX) THEN
X                   longchar:=BConIn(AUX);
X                   t:=BITSET(longchar);
X                   EXCL(t,8);
X                   char:=CHAR(t);
X                   RETURN;
X                END;
X        IF ((GetTime() DIV 20)>WaitTime)
X        OR ((GetTime() DIV 20)=WaitTime) THEN
X           INC(ticks);
X           WaitTime:=LONGCARD(time)+(GetTime() DIV 20);
X           IF ticks=2 THEN
X              char:=CHAR(255);
X              RETURN;
X           END;
X        END;
X        END; (* loop *)
XEND             readModem;
X
XPROCEDURE       flushinput();
XVAR             char    :       LONGCARD;
XBEGIN
X        WHILE BConStat(AUX) DO
X              char:=BConIn(AUX);
X        END;
XEND             flushinput;
X
XPROCEDURE       sendchar(char: CHAR);
XBEGIN
X        BConOut(AUX,char);
XEND             sendchar;
X
XPROCEDURE       xmodemrec(filename: ARRAY OF CHAR): BOOLEAN;
XVAR             sectnum,sectcurr,sectcomp,fd            :       INTEGER;
X                errors                                  :       INTEGER;
X                firstchar                               :       CHAR;
X                errorflag,goodcheck,crc1,crc2           :       BOOLEAN;
X                checksum,j,bufptr                       :       CARDINAL;
X                b                                       :       LONGCARD;
X                bufr                                    :       ARRAY [0..BUFSIZ]
OF CHAR;
XBEGIN
X        IF rec OR snd THEN
```

```
X           WriteLn;
X           WriteString(" XMODEM already active! ");
X           WriteLn;
X           RETURN FALSE;
X         END;
X         rec:=TRUE;
X         mdmini();
X         SFirst(filename,0,result);
X         IF result=0 THEN
X           WriteLn;
X           WriteString(filename);
X           WriteString(" already exists! ");
X           WriteLn;
X           errorbells;
X           xmodemerror:=(-1);
X           rec:=FALSE;
X           RETURN FALSE;
X         END;
X         Create(filename,0,fd);
X         IF fd<0 THEN
X           WriteLn;
X           WriteString("GEMDOS ERROR # ");
X           WriteInt(fd,2);
X           WriteLn;
X           xmodemerror:=fd;
X           IF Close(fd) THEN END;
X           rec:=FALSE;
X           RETURN FALSE;
X       ELSE
X           Assign(xfrname,filename);
X           WriteString(" receiving ");
X           WriteString(filename);
X           WriteLn;
X         END;
X(*         crcmode:=TRUE; *)
X         sectnum:=0; errors:=0; bufptr:=0;
X         flushinput();
X         IF crcmode THEN
X           sendchar(C);
X         ELSE
X           sendchar(NAK);
X         END;
X
X         WHILE (firstchar#EOT) AND (errors#ERRORMAX) DO
X               errorflag:=FALSE;
X
X               t1:=timerset(50);
X               REPEAT
X                 readModem(readchar,5);
X                 firstchar:=readchar;
X                 IF xmodemabort THEN
X                    IF Close(fd) THEN END;
X                    xmodemerror:=(-11);
```

```
X                  rec:=FALSE;
X                  RETURN FALSE;
X               END;
X               IF timeup(t1) THEN
X                  t1:=timerset(50);
X                  flushinput();
X                  IF crcmode THEN
X                     sendchar(C);
X                  ELSE
X                     sendchar(NAK);
X                  END;
X                  IF errors>ERRORMAX DIV 2 THEN crcmode:=NOT crcmode END;
X                  INC(errors);
X                  IF errors>ERRORMAX THEN
X                     IF Close(fd) THEN END;
X                     xmodemerror:=(-1);
X                     rec:=FALSE;
X                     RETURN FALSE;
X                  END;
X               END;
X            UNTIL (firstchar=SOH) OR (firstchar=EOT);
X
X            IF firstchar=SOH THEN
X               readModem(readchar,5);
X               sectcurr:=INTEGER(readchar);
X               readModem(readchar,5);
X               sectcomp:=INTEGER(readchar);
X               IF sectcurr+sectcomp=255 THEN
X                  IF sectcurr=IAnd255(sectnum+1) THEN
X                     checksum:=0;
X                     FOR j:=bufptr TO bufptr+SECSIZ-1 DO
X                         IF xmodemabort THEN
X                            IF Close(fd) THEN END;
X                            xmodemerror:=(-11);
X                            rec:=FALSE;
X                            RETURN FALSE;
X                         END;
X                         readModem(readchar,5);
X                         bufr[j]:=readchar;
X                         IF crcmode THEN
X                            checksum:=crcupdate(checksum,bufr[j]);
X                         ELSE
X                            checksum:=checksum+CARDINAL(bufr[j]);
X                         END;
X                     END; (* for *)
X
X                  IF crcmode THEN
X                     crc1:=FALSE; crc2:=FALSE;
X                     (*
X                     FOR j:=bufptr TO bufptr+SECSIZ-1 DO
X                         checksum:=crcupdate(checksum,bufr[j]);
X                     END;
X                     *)
```

```
X                     checksum:=crcfinish(checksum);
X                     readModem(readchar,5);
X                     IF readchar=CHAR(IAnd255(CARDINAL(WShr(checksum,8)))) THEN
X                        crc1:=TRUE;
X                     END;
X                     readModem(readchar,5);
X                     IF readchar=CHAR(IAnd255(checksum)) THEN
X                        crc2:=TRUE;
X                     END;
X                     IF crc1 AND crc1 THEN
X                        goodcheck:=TRUE;
X                     ELSE
X                        goodcheck:=FALSE;
X                     END;
X                  ELSE
X                  (*
X                     FOR j:=bufptr TO bufptr+SECSIZ-1 DO
X                         checksum:=checksum+CARDINAL(bufr[j]);
X                     END;
X                  *)
X                     readModem(readchar,5);
X                     IF checksum=CARDINAL(readchar) THEN
X                        goodcheck:=TRUE;
X                     ELSE
X                        goodcheck:=FALSE;
X                     END;
X                  END;
X
X
X                       IF goodcheck THEN
X                          INC(mdmPacketsReceived);
X                          errors:=0;
X                          INC(sectnum);
X                          bufptr:=bufptr+SECSIZ;
X                          mdmBytesXferred:=mdmBytesXferred+SECSIZ;
X                          IF bufptr=BUFSIZ THEN
X                             bufptr:=0;
X                             b:=BUFSIZ;
X                             Write(fd,b,ADR(bufr));
X                          END;
X(*     this is for error checking for the write *)
X                          flushinput;
X                          sendchar(ACK);
X                       ELSE
X                          INC(mdmBadPackets);
X                          errorflag:=TRUE;
X                       END; (* if *)
X                     ELSE
X                       IF sectnum=IAnd255(sectnum) THEN
X                          flushinput;
X                          sendchar(ACK);
X                       ELSE
X                          INC(mdmBadPackets);
```

```
X                        errorflag:=TRUE;
X                     END;
X                  END; (* if *)
X               ELSE
X                  INC(mdmBadPackets);
X                  errorflag:=TRUE;
X               END; (* if *)
X            IF errorflag THEN
X               INC(errors);
X               flushinput;
X               IF crcmode THEN
X                  sendchar(C);
X               ELSE
X                  sendchar(NAK);
X               END;
X            END;
X            END; (* if *)
X        END; (* while *)
X        IF (firstchar=EOT) AND (errors< ERRORMAX) THEN
X           sendchar(ACK);
X           b:=LONGCARD(bufptr);
X           Write(fd,b,ADR(bufr));
X           IF Close(fd) THEN END;
X           xmodemerror:=0;
X           rec:=FALSE;
X           ok:=TRUE;
X           RETURN TRUE;
X        END;
X        IF Close(fd) THEN END;
X        xmodemerror:=(-1);
X        rec:=FALSE;
X        RETURN FALSE;
XEND             xmodemrec;
X
XPROCEDURE       xmodemsnd(filename: ARRAY OF CHAR): BOOLEAN;
XVAR             sectnum,attempts,fd          :   INTEGER;
X                checksum,j,bufptr            :   CARDINAL;
X                readchar,c,nak               :   CHAR;
X                b                            :   LONGCARD;
X                dtaAdr                       :   ADDRESS;
X                bufr                         :   ARRAY [0..BUFSIZ] OF CHAR;
XBEGIN
X        IF rec OR snd THEN
X           WriteLn;
X           WriteString(" XMODEM already active! ");
X           WriteLn;
X           RETURN FALSE;
X        END;
X        snd:=TRUE;
X(*        crcmode:=TRUE; *)
X        mdmini();
X        setbuffer(ADR(bufr),BUFSIZ,0c); (* clear buffer *)
X        Open(filename,0,fd);
```

```
X        IF fd<0 THEN
X           WriteLn;
X           WriteString("GEMDOS ERROR # ");
X           WriteInt(fd,2);
X           WriteLn;
X           xmodemerror:=fd;
X           IF Close(fd) THEN END;
X           snd:=FALSE;
X           RETURN FALSE;
X        ELSE
X           GetDTA(dtaAdr);
X           SFirst(filename,0,result);
X           filesize:=dtaAdr+26;
X           endblk:=INTEGER(((filesize^+127) DIV 128)+1);
X           Assign(xfrname,filename);
X           WriteString(" sending ");
X           WriteString(filename );
X           WriteInt(endblk,5);
X           WriteString(" block(s)");
X           WriteLn;
X        END;
X        attempts:=0;
X        sectnum:=1;
X        j:=0;
X        IF crcmode THEN
X           nak:=C;
X        ELSE
X           nak:=NAK;
X        END;
X        readModem(readchar,5);
X        c:=readchar;
X        WHILE (c#nak) AND (j<ERRORMAX) DO
X              readModem(readchar,20);
X              c:=readchar;
X              IF j> ERRORMAX DIV 2 THEN
X                 crcmode:=NOT crcmode;
X                 IF crcmode THEN
X                    nak:=C;
X                 ELSE
X                    nak:=NAK;
X                 END;
X              END;
X              INC(j);
X              IF xmodemabort OR (j=ERRORMAX) THEN
X                 IF Close(fd) THEN END;
X                 xmodemerror:=(-11);
X                 snd:=FALSE;
X                 RETURN FALSE;
X              END;
X        END; (* while *)
X        flushinput;
X        WHILE (endblk#0) AND (attempts#RETRYMAX) DO
X              setbuffer(ADR(bufr),BUFSIZ,CTRLZ);
```

```
X              b:=BUFSIZ;
X              Read(fd,b,ADR(bufr));
X              bufptr:=0;
X              REPEAT
X                attempts:=0;
X                REPEAT
X                  IF xmodemabort THEN
X                     IF Close(fd) THEN END;
X                     xmodemerror:=(-11);
X                     snd:=FALSE;
X                     RETURN FALSE;
X                  END;
X                  sendchar(SOH);
X                  sendchar(CHAR(IAnd255(sectnum)));
X                  sendchar(CHAR(255-IAnd255(sectnum)));
X                  checksum:=0;
X                  writeModem(ADR(bufr[bufptr]),SECSIZ);
X                  IF crcmode THEN
X                     FOR j:=bufptr TO bufptr+SECSIZ-1 DO
X                         checksum:=crcupdate(checksum,bufr[j]);
X                     END;
X                     checksum:=crcfinish(checksum);
X                     sendchar(CHAR(IAnd255(CARDINAL(WShr(checksum,8)))));
X                     sendchar(CHAR(IAnd255(checksum)));
X                  ELSE
X                     FOR j:=bufptr TO bufptr+SECSIZ-1 DO
X                         checksum:=checksum+CARDINAL(bufr[j]);
X                     END;
X                     sendchar(CHAR(IAnd255(checksum)));
X                  END;
X                  flushinput;
X                  INC(mdmPacketsSent);
X                  INC(attempts);
X                  readModem(readchar,mtimeout);
X                  c:=readchar;
X                  IF c#ACK THEN
X                     INC(mdmNakedPackets);
X                  END;
X                UNTIL (c=ACK) OR (attempts=RETRYMAX);
X                IF attempts#RETRYMAX THEN
X                   DEC(endblk);
X                   bufptr:=bufptr+SECSIZ;
X                   mdmBytesXferred:=mdmBytesXferred+SECSIZ;
X                   INC(sectnum);
X                END;
X              UNTIL (bufptr=BUFSIZ) OR (attempts=RETRYMAX) OR (endblk=0);
X        END; (* while *)
X
X        IF Close(fd) THEN END;
X        IF attempts=RETRYMAX THEN
X           xmodemerror:=(-1);
X           snd:=FALSE;
X           RETURN FALSE;
```

```
X        ELSE
X           attempts:=0;
X           REPEAT
X                sendchar(EOT);
X                INC(attempts);
X                readModem(readchar,5);
X                c:=readchar;
X           UNTIL (c=ACK) OR (attempts=RETRYMAX);
X           xmodemerror:=0;
X           snd:=FALSE;
X           ok:=TRUE;
X           RETURN TRUE;
X        END;
XEND             xmodemsnd;
X
XBEGIN
XEND XMODEM.
SHAR_EOF
chmod 0600 XMODEM.MOD || echo "restore of XMODEM.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > XMODEMR.MOD &&
X(*
X        1       check xmodem status
X        2       change error-check method
X        99      abort xmodem
X*)
XMODULE  xmodemr;
XFROM    SYSCALL IMPORT  EnableSpint,DisableSpint;
XFROM    GEMDOS  IMPORT  OldTerm,ConOut;
XFROM    GEMX    IMPORT  BasePageAddress;
XFROM    SYSTEM  IMPORT  ADR,ADDRESS;
XFROM    XMODEM  IMPORT  xmodemrec,xmodemabort,crcmode,xmodemerror,
X                        xmodemstat;
XVAR     ok                              : BOOLEAN;
X        spintcmd                        : ARRAY [0..2] OF LONGCARD;
X        cmd                             : ARRAY [0..81] OF CHAR;
X        i,delay                         : CARDINAL;
X
XPROCEDURE       run;
XBEGIN
X        IF spintcmd[0]=1 THEN xmodemstat END;
X        IF spintcmd[0]=99 THEN xmodemabort:=TRUE END;
XEND             run;
X
XBEGIN
X        IF EnableSpint(1,run,ADR(spintcmd)) THEN
X           spintcmd[1]:=LONGCARD(xmodemstat);
X           crcmode:=TRUE;
X           FOR i:=1 TO ORD(BasePageAddress^.CmdLine[0])+1 DO
X              cmd[i-1]:=BasePageAddress^.CmdLine[i];
X           END;
X           cmd[ORD(BasePageAddress^.CmdLine[0])]:=0c;
X           IF BasePageAddress^.CmdLine[0]#0c THEN
X              IF cmd[0]='-' THEN
```

```
X                 crcmode:=FALSE;
X                 FOR i:=0 TO ORD(BasePageAddress^.CmdLine[0]) DO
X                    cmd[i]:=cmd[i+1];
X                 END;
X              END;
X           IF xmodemrec(cmd) THEN
X              ConOut(7c);
X           ELSE
X              FOR i:=0 TO 3 DO
X                 FOR delay:=0 TO 10000 DO END;
X                 ConOut(7c);
X              END;
X           END; (* get the file *)
X           END;
X           DisableSpint(1);
X           OldTerm;
X        END;
XEND     xmodemr.
X
SHAR_EOF
chmod 0600 XMODEMR.MOD || echo "restore of XMODEMR.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > XMODEMT.MOD &&
X(*
X        1       check xmodem status
X        2       change error-check method
X        99      abort xmodem
X*)
XMODULE  xmodemt;
XFROM    SYSCALL IMPORT  EnableSpint,DisableSpint;
XFROM    GEMDOS  IMPORT  OldTerm,ConOut;
XFROM    GEMX    IMPORT  BasePageAddress;
XFROM    SYSTEM  IMPORT  ADR,ADDRESS;
XFROM    XMODEM  IMPORT  xmodemsnd,xmodemabort,crcmode,xmodemerror,
X                        xmodemstat;
XVAR     ok                              : BOOLEAN;
X        spintcmd                        : ARRAY [0..2] OF LONGCARD;
X        cmd                             : ARRAY [0..81] OF CHAR;
X        i,delay                         : CARDINAL;
X
XPROCEDURE       run;
XBEGIN
X        IF spintcmd[0]=1 THEN xmodemstat END;
X        IF spintcmd[0]=99 THEN xmodemabort:=TRUE END;
XEND             run;
X
XBEGIN
X        IF EnableSpint(1,run,ADR(spintcmd)) THEN
X           spintcmd[1]:=LONGCARD(xmodemstat);
X           crcmode:=TRUE;
X           FOR i:=1 TO ORD(BasePageAddress^.CmdLine[0])+1 DO
X              cmd[i-1]:=BasePageAddress^.CmdLine[i];
X           END;
X           cmd[ORD(BasePageAddress^.CmdLine[0])]:=0c;
```

```
X           IF BasePageAddress^.CmdLine[0]#0c THEN
X              IF cmd[0]='-' THEN
X                 crcmode:=FALSE;
X                 FOR i:=0 TO ORD(BasePageAddress^.CmdLine[0]) DO
X                    cmd[i]:=cmd[i+1];
X                 END;
X              END;
X           IF xmodemsnd(cmd) THEN
X              ConOut(7c);
X           ELSE
X              FOR i:=0 TO 3 DO
X                 FOR delay:=0 TO 10000 DO END;
X                 ConOut(7c);
X              END;
X           END; (* send the file *)
X           END;
X           DisableSpint(1);
X           OldTerm;
X        END;
XEND     xmodemt.
X
SHAR_EOF
chmod 0600 XMODEMT.MOD || echo "restore of XMODEMT.MOD fails"
sed 's/^X//' << 'SHAR_EOF' > XS.MOD &&
XMODULE  xs;
XFROM    SYSCALL         IMPORT  SpintInfo;
XFROM    GEMDOS          IMPORT  Term;
XFROM    Terminal        IMPORT  WriteString,WriteLn;
XVAR     spintcmd        :       POINTER TO ARRAY [0..2] OF LONGCARD;
X        XS              :       PROC;
X        ok              :       BOOLEAN;
XBEGIN
X        IF SpintInfo(1,spintcmd) THEN
X           XS:=PROC(spintcmd^[1]);
X           XS();
X        ELSE
X           WriteLn;
X           WriteString("Background xmodem not running");
X           WriteLn;
X           ok:=Term(-1);
X        END;
X        ok:=Term(0);
XEND     xs.
SHAR_EOF
chmod 0600 XS.MOD || echo "restore of XS.MOD fails"
rm -f s2_seq_.tmp
echo "You have unpacked the last part"
exit 0


From -1073908949849986353
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,1c4a3d4f8f04ef9c,start
```

X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-01-18 01:39:37 PST
Path: utzoo!attcan!uunet!lll-winken!ncis.llnl.gov!ncis!helios.ee.lbl.gov!pasteur!
agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i089: limiter -- Limit uppermost hard disk device letter
Keywords: GFA BASIC
Message-ID: <5994@saturn.ucsc.edu>
Date: 18 Jan 89 09:39:37 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 38
Approved: koreth@ssyx.ucsc.edu

Submitted-by: holzer@ztivax.siemens.com (Walter Holzer)
Posting-number: Volume 1, Issue 89
Archive-name: limiter

```
' ------------------------------------------------------------------------
' LIMITER, a Public Domain program by MaDom, <P> 26. Dec. 1988
' Version 1.0 developed in GfA-BASIC 2.0 because the compiler V3 isn't
' available (Oh sh..)
'
' Description:
' If you're using the driver's from Supra or ICD, all drives (A through P)
' are activated. This seems a little stupid, so this program was developed.
' You've to put the program in the folder "AUTO" of your boot device. Then
' you've to rename the program to LIMITERx.PRG, where x means the character
' of your last harddisk partition, eg. LIMITERI.PRG if your last partition
' will be "I:".
'
' CAUTION: The program won't check your inputs, so be careful.
' ------------------------------------------------------------------------
' set dta

Void Gemdos(26,L:Basepage+128)
' &H446 means the boot device, so every boot device will work
Filename$=Chr$(Dpeek(&H446)+65)+":\AUTO\LIMITER?.PRG"+Chr$(0)
' Scan for the name of the program
E%=Gemdos(78,L:Varptr(Filename$),-1)
' Yes, we've got it
If E%=0 Then
  Limit=Peek(Basepage+165)-64
  ' Poke the new number into _drvbits
  Sdpoke &H4C4,(2^Limit)-1
  ' A little bit of Copyright

  Print "LIMITER, a Public Domain program by MaDOM, <P> 26. Dec. 1988"
Endif
End
```

From 712480488928026268
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,11f1a798917a4c36,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-01-18 21:53:41 PST
Path: utzoo!attcan!uunet!lll-winken!ncis.llnl.gov!ncis!helios.ee.lbl.gov!pasteur!
ucbvax!agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i090: d110edit -- Patch editor for D110 synthesizer
Keywords: shar, C
Message-ID: <6022@saturn.ucsc.edu>
Date: 19 Jan 89 05:53:41 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1949
Approved: koreth@ssyx.ucsc.edu

Submitted-by: kurth@sco.com (Kurt Hutchinson)
Posting-number: Volume 1, Issue 90
Archive-name: d110edit

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      keys.c
#      memcpy.s
#      mt.c
#      mt.h
#      mtscrn.c
#      mtscrn.h
#      mttype.h
#      osbind.h
#      patch.c
#      setuptmp.c
#      src.c
#      stdio.h
#      system.c
#      timbre.c
#      timbretm.c
#      tone.c
#      top.c
#      makefile
#      d110.doc
#
echo "x - extracting keys.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > keys.c &&
X#include <osbind.h>
X#include "mt.h"
X
X/*
X * keyboard to midi translator
```

```
X */
X
Xunsigned char midichan;
X
X/* note on, note #, note intensity */
Xunsigned char mtnote[3] = { 0x92, 0x00, 0x7d };
Xunsigned char mtnoteoff[3] = { 0xb2, 0x7b, 0x00 };
X
Xmtkeys(l)
Xlong l;
X{
X    int k;
X    int in;
X
X    k = l>>16;
X    if (k == 0x62) {
X         help();
X         return;
X    }
X    mtnote[1] = 0;
X    in = 0;
X    k = l&0x7f;
X    switch (k) {
X    case '?':
X         help();
X         return;
X    case '1':
X         in = 0x07;
X         break;
X    case '2':
X         in = 0x17;
X         break;
X    case '3':
X         in = 0x37;
X         break;
X    case '4':
X         in = 0x47;
X         break;
X    case '5':
X         in = 0x57;
X         break;
X    case '6':
X         in = 0x67;
X         break;
X    case '7':
X         in = 0x77;
X         break;
X    case '8':
X         in = 0x7f;
X         break;
X    case 'q':
X         mtnote[1] = 24;         /* C */
X         break;
```

```
X    case 'w':
X         mtnote[1] = 28;        /* E */
X         break;
X    case 'e':
X         mtnote[1] = 31;        /* G */
X         break;
X    case 'r':
X         mtnote[1] = 36;        /* C */
X         break;
X    case 't':
X         mtnote[1] = 40;        /* E */
X         break;
X    case 'y':
X         mtnote[1] = 43;        /* G */
X         break;
X    case 'u':
X         mtnote[1] = 48;        /* C */
X         break;
X    case 'i':
X         mtnote[1] = 52;        /* E */
X         break;
X    case 'o':
X         mtnote[1] = 55;        /* G */
X         break;
X    case 'p':
X         mtnote[1] = 60;        /* C */
X         break;
X    case '[':
X         mtnote[1] = 64;        /* E */
X         break;
X    case ']':
X         mtnote[1] = 67;        /* G */
X         break;
X    case 'a':
X         mtnote[1] = 72;        /* C */
X         break;
X    case 's':
X         mtnote[1] = 76;        /* E */
X         break;
X    case 'd':
X         mtnote[1] = 79;        /* G */
X         break;
X    case 'f':
X         mtnote[1] = 84;        /* C */
X         break;
X    case 'g':
X         mtnote[1] = 88;        /* E */
X         break;
X    case 'h':
X         mtnote[1] = 91;        /* G */
X         break;
X    case 'j':
X         mtnote[1] = 96;        /* C */
```

```
X         break;
X    case 'k':
X         mtnote[1] = 100;   /* E */
X         break;
X    case 'l':
X         mtnote[1] = 103;   /* G */
X         break;
X    case ';':
X         mtnote[1] = 108;   /* C */
X         break;
X    case ' ':
X         mtnoteoff[0] &= 0xf0;
X         mtnoteoff[0] |= midichan;
X         Midiws(3, mtnoteoff);
X         return;
X    }
X    if (in)
X         mtnote[2] = in;
X    if (mtnote[1]) {
X         mtnote[0] &= 0xf0;
X         mtnote[0] |= midichan;
X         Midiws(3, mtnote);
X    }
X}
X
Xhelp()
X{
X    kclear();
Xprintf("F1 to F10 are -16, -8, -4, -2, -1, +1, +2, +4, +8, +16 respectively\n");
Xprintf("Arrow keys select field\n");
Xprintf("Escape goes to previous level\n");
Xprintf("Enter takes selected action or previous level\n");
Xprintf("\nMidi test Bed only works on numeric fields\n");
Xprintf("keys 1 to 8 select intensity\n");
Xprintf("keys q-] and a-; are all notes in the C arpeggio\n");
X    printf("\n\tenter any key to continue\n");
X    while (Bconstat(CON));
X         Bconin(CON);
X    drawscrn();
X}
SHAR_EOF
chmod 0664 keys.c || echo "restore of keys.c fails"
echo "x - extracting memcpy.s (Text)"
sed 's/^X//' << 'SHAR_EOF' > memcpy.s &&
X    .shri
X    .globl    memcpy_
Xmemcpy_:
X    link a6, $0
X    movea.l   8(a6), a0
X    movea.l   12(a6), a1
X    clr.l     d0
X    move.w    16(a6), d0
X    cmpi.l    $15, d0
```

```
X    bgt  long
Xtop:
X    subq.l   $1, d0
Xloop:
X    move.b   (a0)+, (a1)+
X    dbf  d0, loop
Xdone:
X    unlk a6
X    rts
X
Xlong:
X    move.l   a0, d1
X    move.l   a1, d2
X    btst $0, d1
X    beq  z1
X    btst $0, d2
X    beq  loop
X    move.b   (a0)+, (a1)+  / both odd
X    subq.l   $1, d0
Xaligned:
X    move.l   d0, d1
X    lsr.l    $2, d1        / long word count
X    subq.l   $1, d1
Xlloop:
X    move.l   (a0)+, (a1)+
X    dbf  d1, lloop
X    andi.l   $3, d0
X    beq  done
X    jmp  top
X
Xz1:
X    btst $0, d2
X    beq  aligned
X    jmp  loop
X SHAR_EOF
chmod 0664 memcpy.s || echo "restore of memcpy.s fails"
echo "x - extracting mt.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > mt.c &&
Xmain()
X{
X    top();
X}
SHAR_EOF
chmod 0664 mt.c || echo "restore of mt.c fails"
echo "x - extracting mt.h (Text)"
sed 's/^X//' << 'SHAR_EOF' > mt.h &&
X#define CON  2
X#define MIDI 3
X#define RQMAX     16400
X
X#define uchar     unsigned char
X#define uint unsigned int
SHAR_EOF
```

```
chmod 0664 mt.h || echo "restore of mt.h fails"
echo "x - extracting mtscrn.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > mtscrn.c &&
X#include <osbind.h>
X#include "mt.h"
X#include "mtscrn.h"
X
Xscreen_t *csp;         /* current screen */
Xint cfld;         /* current field index */
Xuchar sbuf[81];         /* ascii field data buffer */
Xint cindex;       /* current index into an ascii field */
X
Xscreen(sp)
Xregister screen_t *sp;
X{
X    csp = sp;
X    drawscrn();
X
X    scrinput();
X}
X
Xscrinput()
X{
X    register screen_t *sp;
X    register field_t *fp;
X    uchar *cp;
X    register int c;
X    long l;
X
X    midiflush();
X    setfld(csp->s_fp);
X    for (;;) {
X         while (Bconstat(CON) == 0) { /* midi pass thru feature */
X              if (Bconstat(MIDI)) {
X                   c = Bconin(MIDI);
X                   Bconout(MIDI, c);
X              }
X         }
X         l = Bconin(CON);   /* we have a character to read */
X         if ((l&0xff) == 0x1b) {/* escape is no write */
X              return(-1);
X         }
X         sp = csp;
X         fp = &sp->s_fp[cfld];
X         c = (l>>16)&0xff;
X         switch (c) {
X         case 0x1c:    /* enter done with screen */
X              (*sp->s_done)();
X              return(0);
X         case 0x4b:    /* left arrow */
X              if (cfld)
X                   fp--;
X              setfld(fp);
```

```
X            continue;
X        case 0x48:    /* up arrow */
X            c = fp->f_y;
X            while (cfld) {
X                fp--;
X                cfld--;
X                if (fp->f_y < c)
X                    break;
X            }
X            while (cfld) {
X                if ((fp-1)->f_y == fp->f_y) {
X                    fp--;
X                    cfld--;
X                }
X                else
X                    break;
X            }
X            setfld(fp);
X            continue;
X        case 0x4d:    /* right arrow */
X            if ((cfld + 1) < sp->s_fieldsiz)
X                fp++;
X            setfld(fp);
X            continue;
X        case 0x50:    /* down arrow */
X            c = fp->f_y;
X            while ((cfld + 1) < sp->s_fieldsiz) {
X                fp++;
X                cfld++;
X                if (fp->f_y > c)
X                    break;
X            }
X            setfld(fp);
X            continue;
X        }
X        if (fp->f_type == 1) { /* ascii field */
X            c = l&0x7f;
X            if ((c >= 'a' && c <= 'z') ||
X                (c >= 'A' && c <= 'Z') ||
X                (c >= '0' && c <= '9') ||
X                (c == '.') || (c == ' ')) {
X                cp = (*sp->s_get)(fp->f_disp);
X                *(cp + cindex) = c;
X                if ((cindex + 1) < fp->f_len)
X                    cindex++;
X                (*sp->s_put)(fp->f_disp);
X            }
X            else {
X                (*sp->s_ukn)(l);
X                continue;
X            }
X        }
X        else {            /* numeric field */
```

```
X             switch (c) {
X             case 0x3b:    /* F1 */
X                  fchange(-16);
X                  break;
X             case 0x3c:    /* F2 */
X                  fchange(-8);
X                  break;
X             case 0x3d:    /* F3 */
X                  fchange(-4);
X                  break;
X             case 0x3e:    /* F4 */
X                  fchange(-2);
X                  break;
X             case 0x3f:    /* F5 */
X                  fchange(-1);
X                  break;
X             case 0x40:    /* F6 */
X                  fchange(1);
X                  break;
X             case 0x41:    /* F7 */
X                  fchange(2);
X                  break;
X             case 0x42:    /* F8 */
X                  fchange(4);
X                  break;
X             case 0x43:    /* F9 */
X                  fchange(8);
X                  break;
X             case 0x44:    /* F10 */
X                  fchange(16);
X                  break;
X             default:
X                  (*sp->s_ukn)(l);
X                  continue;
X             }
X         }
X         drawfield(fp);
X         ksetcrsr(fp->f_x + cindex, fp->f_y);
X    }
X}
X
Xfchange(change)
X{
X    register screen_t *sp;
X    register field_t *fp;
X    register uchar *cp;
X    int c;
X
X    sp = csp;
X    fp = &sp->s_fp[cfld];
X    cp = (*sp->s_get)(fp->f_disp);
X    c = *cp + change;
X    if (c < 0)
```

```
X         c = 0;
X    if (c > fp->f_max)
X         c = fp->f_max;
X    *cp = c;
X    (*sp->s_put)(fp->f_disp);
X}
X
Xsetfld(fp)
Xregister field_t *fp;
X{
X    cindex = 0;
X    cfld = fp - csp->s_fp;
X    ksetcrsr(fp->f_x, fp->f_y);
X}
X
Xdrawscrn()
X{
X    kclear();
X    drawdata();
X    drawflds();
X}
X
Xdrawdata()
X{
X    register screen_t *sp;
X    register data_t *dp;
X    register int i;
X
X    sp = csp;
X    dp = sp->s_dp;
X    for (i = 0; i < sp->s_datasiz; i++) {
X         ksetcrsr(dp->d_x, dp->d_y);
X         kouts(dp->d_cp);
X         dp++;
X    }
X}
X
Xdrawflds()
X{
X    register screen_t *sp;
X    register field_t *fp;
X    register int i;
X
X    sp = csp;
X    fp = sp->s_fp;
X    for (i = 0; i < sp->s_fieldsiz; i++) {
X         drawfield(fp);
X         fp++;
X    }
X}
X
Xdrawfield(fp)
Xregister field_t *fp;
```

```
X{
X    register screen_t *sp;
X    uchar *cp;
X    int t;
X    int i;
X
X    sp = csp;
X    ksetcrsr(fp->f_x, fp->f_y);
X    switch (fp->f_type) {
X    case 2:            /* binary */
X         cp = (*sp->s_get)(fp->f_disp);
X         t = *cp + fp->f_offset;
X         cp = sbuf + fp->f_len - 1;
X         for (i = 0; i < fp->f_len; i++) {
X              if ((t & 1) == 0)
X                   *cp-- = '0';
X              else
X                   *cp-- = '1';
X              t >>= 1;
X         }
X         sbuf[fp->f_len] = 0;
X         break;
X    case 1:            /* ascii */
X         cp = (*sp->s_get)(fp->f_disp);
X         memcpy(cp, sbuf, fp->f_len);
X         sbuf[fp->f_len] = 0;
X         pad(sbuf, fp->f_len);
X         break;
X    case 0:            /* decimal */
X         cp = (*sp->s_get)(fp->f_disp);
X         t = *cp + fp->f_offset;
X         sprintf(sbuf, "%d", t);
X         sbuf[fp->f_len] = 0;
X         pad(sbuf, fp->f_len);
X    }
X    kouts(sbuf);
X}
X
Xkclear()
X{
X    Bconout(CON, 0x1b);
X    Bconout(CON, 'E');
X}
X
Xksetcrsr(x, y)
X{
X    Bconout(CON, 0x1b);
X    Bconout(CON, 'Y');
X    Bconout(CON, y + 0x20);
X    Bconout(CON, x + 0x20);
X}
X
Xkoutc(c)
```

```
X{
X    Bconout(CON, c);
X}
X
Xkouts(cp)
Xregister uchar *cp;
X{
X    while (*cp)
X         Bconout(CON, *cp++);
X}
X
Xpad(cp, len)
Xregister uchar *cp;
X{
X    int i;
X
X    cp[len] = 0;
X    i = strlen(cp);
X    while (i < len)
X         cp[i++] = ' ';
X}
SHAR_EOF
chmod 0664 mtscrn.c || echo "restore of mtscrn.c fails"
echo "x - extracting mtscrn.h (Text)"
sed 's/^X//' << 'SHAR_EOF' > mtscrn.h &&
X/*
X * defines for the two screen structures
X */
X
Xstruct datadef {
X    uchar    d_y;      /* x coordinate for string */
X    uchar    d_x;      /* y coordinate for string */
X    uchar    *d_cp;         /* the string itself */
X};
X
Xtypedef struct datadef data_t;
X
Xstruct fielddef {
X    uchar    f_y;      /* x coordinate of field */
X    uchar    f_x;      /* y coordinate of field */
X    uchar    f_type;        /* 0 is number, non-zero is ascii */
X    uchar    f_len;         /* field length */
X    uchar    f_max;         /* maximum value */
X    char f_offset; /* offset to be added before display */
X    int  f_disp;        /* displacement of data */
X};
X
Xtypedef struct fielddef field_t;
X
Xstruct screendef {
X    data_t   *s_dp;         /* data pointer */
X    int  s_datasiz;    /* size of data array */
X    field_t *s_fp;          /* field pointer */
```

```
X    int  s_fieldsiz;   /* size of field array */
X    uchar    *(*s_get)();  /* get data function */
X    int  (*s_put)();   /* put data function */
X    int  (*s_ukn)();   /* unknown character function */
X    int  (*s_done)();  /* end of screen routine */
X};
X
Xtypedef struct screendef screen_t;
SHAR_EOF
chmod 0664 mtscrn.h || echo "restore of mtscrn.h fails"
echo "x - extracting mttype.h (Text)"
sed 's/^X//' << 'SHAR_EOF' > mttype.h &&
X/*
X * defines for the source type structure
X */
X
Xstruct mtypedef {
X    int  t_type;       /* source type */
X    int  t_psiz;       /* physical size (displacement between adjacent
X                       entries */
X    int  t_vsiz;       /* virtual size (size of read and write
X                       requests */
X    uchar    t_base[3];    /* base memory address in mt32 */
X    int  t_max;        /* max entries in mt32 memory area */
X    uchar    t_filesuf[6]; /* file suffix for this type */
X};
X
Xtypedef struct mtypedef type_t;
X
X#define T_TIMBRE  0    /* timbre */
X#define T_TIMBRETMP   1    /* timbre tmp */
X#define T_TONE        2    /* tone memory */
X#define T_TONETMP 3    /* tone tmp memory */
X#define T_SYSTEM  4    /* system memory area */
X#define T_SETUPTMP    5    /* setup tmp area */
X#define T_PATCH       6    /* patch memory */
SHAR_EOF
chmod 0664 mttype.h || echo "restore of mttype.h fails"
echo "x - extracting osbind.h (Text)"
sed 's/^X//' << 'SHAR_EOF' > osbind.h &&
X/**
X*
X* This file contains macro definitions for use with the Atari specific
X* functions gemdos,bios and xbios
X*
X**/
X
Xextern long bios();
Xextern long xbios();
Xextern long gemdos();
X
X/* GEMDOS functions (trap #1) */
X
```

```
X#define Pterm0()      gemdos(0x0)
X#define Cconin()      gemdos(0x1)
X#define Cconout(a)         gemdos(0x2,a)
X#define Cauxin()      gemdos(0x3)
X#define Cauxout(a)         gemdos(0x4,a)
X#define Cprnout(a)         gemdos(0x5,a)
X#define Crawio(a)     gemdos(0x6,a)
X#define Crawcin()     gemdos(0x7)
X#define Cnecin()      gemdos(0x8)
X#define Cconws(a)     gemdos(0x9,a)
X#define Cconrs(a)     gemdos(0x0a,a)
X#define Cconis()      gemdos(0x0b)
X#define Dsetdrv(a)         gemdos(0x0e,a)
X#define Cconos()      gemdos(0x10)
X#define Cprnos()      gemdos(0x11)
X#define Cauxis()      gemdos(0x12)
X#define Cauxos()      gemdos(0x13)
X#define Dgetdrv()     gemdos(0x19)
X#define Fsetdta(a)         gemdos(0x1a,a)
X#define Super(a)      gemdos(0x20,a)
X#define Tgetdate()         gemdos(0x2a)
X#define Tsetdate(a)        gemdos(0x2b,a)
X#define Tgettime()         gemdos(0x2c)
X#define Tsettime(a)        gemdos(0x2d,a)
X#define Fgetdta()     gemdos(0x2f)
X#define Sversion()         gemdos(0x30)
X#define Ptermres(a,b)      gemdos(0x31,a,b)
X#define Dfree(a,b)         gemdos(0x36,a,b)
X#define Dcreate(a)         gemdos(0x39,a)
X#define Ddelete(a)         gemdos(0x3a,a)
X#define Dsetpath(a)        gemdos(0x3b,a)
X#define Fcreate(a,b)       gemdos(0x3c,a,b)
X#define Fopen(a,b)         gemdos(0x3d,a,b)
X#define Fclose(a)     gemdos(0x3e,a)
X#define Fread(a,b,c)       gemdos(0x3f,a,b,c)
X#define Fwrite(a,b,c)      gemdos(0x40,a,b,c)
X#define Fdelete(a)         gemdos(0x41,a)
X#define Fseek(a,b,c)       gemdos(0x42,a,b,c)
X#define Fattrib(a,b,c)     gemdos(0x43,a,b,c)
X#define Fdup(a)            gemdos(0x45,a)
X#define Fforce(a,b)        gemdos(0x46,a,b)
X#define Dgetpath(a,b)      gemdos(0x47,a,b)
X#define Malloc(a)     gemdos(0x48,a)
X#define Mfree(a)      gemdos(0x49,a)
X#define Mshrink(a,b)       gemdos(0x4a,0,a,b)   /* NOTE: Null parameter added */
X#define Pexec(a,b,c,d)     gemdos(0x4b,a,b,c,d)
X#define Pterm(a)      gemdos(0x4c,a)
X#define Fsfirst(a,b)       gemdos(0x4e,a,b)
X#define Fsnext()      gemdos(0x4f)
X#define Frename(a,b,c)     gemdos(0x56,a,b,c)
X#define Fdatime(a,b,c)     gemdos(0x57,a,b,c)
X
X/* BIOS functions (trap #13) */
```

```
X
X#define  Getmpb(a)     bios(0,a)
X#define Bconstat(a)        bios(1,a)
X#define Bconin(a)     bios(2,a)
X#define Bconout(a,b)       bios(3,a,b)
X#define Rwabs(a,b,c,d,e)   bios(4,a,b,c,d,e)
X#define Setexc(a,b)        bios(5,a,b)
X#define  Tickcal()     bios(6)
X#define  Getbpb(a)     bios(7,a)
X#define Bcostat(a)         bios(8,a)
X#define Mediach(a)         bios(9,a)
X#define Drvmap()      bios(10)
X#define Getshift(a)        bios(11,a)
X
X/* XBIOS functions (trap #14) */
X
X#define Initmous(a,b,c)         (void) xbios(0,a,b,c)
X#define Physbase()              (char *) xbios(2)
X#define Logbase()               (char *) xbios(3)
X#define Getrez()                (int) xbios(4)
X#define Setscreen(a,b,c)        (void) xbios(5,a,b,c)
X#define Setpallete(a)           (void) xbios(6,a)
X#define Setcolor(a,b)           (int) xbios(7,a,b)
X#define Floprd(a,b,c,d,e,f,g)   (int) xbios(8,a,b,c,d,e,f,g)
X#define Flopwr(a,b,c,d,e,f,g)   (int) xbios(9,a,b,c,d,e,f,g)
X#define  Flopfmt(a,b,c,d,e,f,g,h,i) (int) xbios(10,a,b,c,d,e,f,g,h,i)
X/* XBIOS function 11 is unused */
X#define Midiws(a,b)        (void) xbios(12,a,b)
X#define Mfpint(a,b)        (void) xbios(13,a,b)
X#define Iorec(a)      (char *) xbios(14,a)
X#define Rsconf(a,b,c,d,e,f)     (void) xbios(15,a,b,c,d,e,f)
X#define Keytbl(a,b,c)      (char *) xbios(16,a,b,c)
X#define Random()      xbios(17)
X#define Protobt(a,b,c,d)   (void) xbios(18,a,b,c,d)
X#define Flopver(a,b,c,d,e,f,g)  (int) xbios(19,a,b,c,d,e,f,g)
X#define Scrdmp()      (void) xbios(20)
X#define Cursconf(a,b)      (int) xbios(21,a,b)
X#define Settime(a)         (void) xbios(22,a)
X#define Gettime()     xbios(23)
X#define Bioskeys()         (void) xbios(24)
X#define Ikbdws(a,b)        (void) xbios(25,a,b)
X#define Jdisint(a)         (void) xbios(26,a)
X#define Jenabint(a)        (void) xbios(27,a)
X#define Giaccess(a,b)      (char) xbios(28,a,b)
X#define Offgibit(a)        (void) xbios(29,a)
X#define Ongibit(a)         (void) xbios(30,a)
X#define Xbtimer(a,b,c,d)   (void) xbios(31,a,b,c,d)
X#define Dosound(a)         (void) xbios(32,a)
X#define Setprt(a)     (int) xbios(33,a)
X#define Kbdvbase()         (char **) xbios(34)
X#define Kbrate(a,b)        (int) xbios(35,a,b)
X#define Prtblk(a)     (int) xbios(36,a)
X#define  Vsync()           (void) xbios(37)
```

```
X#define Supexec(a)         (unsigned long) xbios(38,a)
X#define Puntaes()     (void) xbios(39)
SHAR_EOF
chmod 0664 osbind.h || echo "restore of osbind.h fails"
echo "x - extracting patch.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > patch.c &&
X#include "mt.h"
X#include "mtscrn.h"
X
Xuchar *mtgetdata();
Xint mtputdata();
Xint mtkeys();
Xint mtdone();
Xextern uchar mtwork[];
X
Xdata_t patchdata[] = {
X    {  1, 28, "D110 Patch Area" },
X    {  3, 10, "Patch Name   [          ]    Rhythm Output Volume 0-100  [   ]" },
X    {  5, 20, "parameters to edit part 1-8 (0 for none) [ ]" },
X    {  7,  1, "Reverb Mode Room 0-1, Hall 2-3, Plate 4, Tap delay 5-7, off 8  [ ]
(0-8)" },
X    {  9,  8, "Reverb Time  [ ]  1-8" },
X    {  9, 34, "Reverb Level  [ ]  0-7" },
X    { 11, 20, "Partial Reserves (0-32), total of 32" },
X    { 13,  0, "ch 1 [  ] ch 2 [  ] ch 3 [  ] ch 4 [  ] ch 5 [  ]" },
X    { 13, 50, "ch 6 [  ] ch 7 [  ] ch 8 [  ]" },
X    { 14,  0, "Rhythm [  ]" },
X    { 15, 16, "Midi Channel For Each Part (1-16, 17 = OFF)" },
X    { 17,  0, "ch 1 [  ] ch 2 [  ] ch 3 [  ] ch 4 [  ] ch 5 [  ]" },
X    { 17, 50, "ch 6 [  ] ch 7 [  ] ch 8 [  ]" },
X    { 18,  0, "Rhythm [  ]" }
X};
X
Xfield_t patchfields[] = {
X/* y    x  type  len  max  offset  disp */
X{  3,  24,    1,  10,   0,      0,    0 },    /* patch name */
X{  3,  68,    0,   3, 100,      0,  127 },    /* rhythm output volume */
X{  5,  62,    0,   1,   8,      0,  128 },    /* parameter to edit */
X{  7,  65,    0,   1,   8,      0,   10 },      /* reverb mode */
X{  9,  22,    0,   1,   7,      1,   11 },      /* reverb time */
X{  9,  49,    0,   1,   7,      0,   12 },    /* reverb level */
X{  13,  6,    0,   2,  32,      0,   13 },    /* reserve ch 1 */
X{  13, 16,    0,   2,  32,      0,   14 },    /* reserve ch 2 */
X{  13, 26,    0,   2,  32,      0,   15 },    /* reserve ch 3 */
X{  13, 36,    0,   2,  32,      0,   16 },    /* reserve ch 4 */
X{  13, 46,    0,   2,  32,      0,   17 },    /* reserve ch 5 */
X{  13, 56,    0,   2,  32,      0,   18 },    /* reserve ch 6 */
X{  13, 66,    0,   2,  32,      0,   19 },    /* reserve ch 7 */
X{  13, 76,    0,   2,  32,      0,   20 },    /* reserve ch 8 */
X{  14,  8,    0,   2,  32,      0,   21 },    /* reserve rhythm */
X{  17,  6,    0,   2,  16,      1,   22 },      /* midi channel 1 */
X{  17, 16,    0,   2,  16,      1,   23 },      /* midi channel 2 */
X{  17, 26,    0,   2,  16,      1,   24 },      /* midi channel 3 */
```

```
X{  17, 36,    0,   2,  16,      1,   25 },      /* midi channel 4 */
X{  17, 46,    0,   2,  16,      1,   26 },      /* midi channel 5 */
X{  17, 56,    0,   2,  16,      1,   27 },      /* midi channel 6 */
X{  17, 66,    0,   2,  16,      1,   28 },      /* midi channel 6 */
X{  17, 76,    0,   2,  16,      1,   29 },      /* midi channel 7 */
X{  18,  8,    0,   2,  16,      1,   30 }       /* midi channel R */
X};
X
Xscreen_t patchscreen = {
X    patchdata,
X    sizeof(patchdata)/sizeof(data_t),
X    patchfields,
X    sizeof(patchfields)/sizeof(field_t),
X    mtgetdata,
X    mtputdata,
X    mtkeys,
X    mtdone
X};
X
Xdata_t paramdata[] = {
X{  1, 30, "D110 Patch Parameter Area" },
X{  4, 5, "Tone Group (a, b, i/c, r)                        [ ]   (0-3)" },
X{  5, 5, "Tone Number                                      [  ]  (0-63)" },
X{  6, 5, "Key Shift                                       [   ] (-24-24)" },
X{  7, 5, "Fine Tune                                       [   ] (-50-50)" },
X{  8, 5, "Bender Range                                    [  ]  (0-24)" },
X{  9, 5, "Assign Mode (single-last,s-first,multi-last,m-f) [ ]   (0-3)" },
X{ 10, 5, "Output Assign (-1 mix, 0 mix+reverb, 1-6         [  ]  (-1-6)" },
X{ 11, 5, "Output Level                                    [   ] (0-100)" },
X{ 12, 5, "Panpot                                          [  ]  (-7-7)" },
X{ 13, 5, "Key range Lower                                 [   ] (0-127)" },
X{ 14, 5, "Key range Upper                                 [   ] (0-127)" }
X};
X
Xfield_t paramfield[] = {
X/*   y   x   type   len   max   offset   disp */
X{    4, 55,     0,   1,     3,       0,     0 },  /* tone name */
X{    5, 55,     0,   2,    63,       0,     1 },  /* tone number */
X{    6, 55,     0,   3,    48,     -24,     2 },  /* key shift */
X{    7, 55,     0,   3,   100,     -50,     3 },  /* fine tune */
X{    8, 55,     0,   2,    24,       0,     4 },  /* bender range */
X{    9, 55,     0,   1,     3,       0,     5 },  /* assign mode */
X{   10, 55,     0,   2,     8,      -1,     6 },  /* output assign */
X{   11, 55,     0,   3,   100,       0,     8 },  /* output level */
X{   12, 55,     0,   2,    14,      -7,     9 },  /* panpot */
X{   13, 55,     0,   3,   127,       0,    10 },  /* key range lower */
X{   14, 55,     0,   3,   127,       0,    11 }        /* key range upper */
X};
X
X#define PDATASIZ  (sizeof(paramdata))
X#define PDATAS        (PDATASIZ/sizeof(data_t))
X#define PFIELDSIZ (sizeof(paramfield))
X#define PFIELDS       (PFIELDSIZ/sizeof(field_t))
```

```
X#define PSCREENSIZ    (sizeof(paramscreen))
X
Xdata_t pdtmp[PDATAS];
Xfield_t pftmp[PFIELDS];
Xscreen_t pstmp;
X
Xscreen_t paramscreen = {
X    paramdata,
X    sizeof(paramdata)/sizeof(data_t),
X    paramfield,
X    sizeof(paramfield)/sizeof(field_t),
X    mtgetdata,
X    mtputdata,
X    mtkeys,
X    mtdone
X};
X
Xspatch()
X{
X    int s;
X    register field_t *fp;
X
X    mtwork[128] = 0;       /* param channel to edit */
X
X    for (;;) {
X        if (screen(&patchscreen) < 0)
X            return;
X        switch (s = mtwork[128]) {
X        case 0:            /* no partial edits */
X            return;
X        case 1:
X        case 2:
X        case 3:
X        case 4:
X        case 5:
X        case 6:
X        case 7:
X        case 8:
X            s--;
X            s *= 0x0c;         /* param data area */
X            s += 0x1f;         /* common data size */
X            memcpy(paramdata, pdtmp, PDATASIZ);
X            memcpy(paramfield, pftmp, PFIELDSIZ);
X            memcpy(&paramscreen, &pstmp, PSCREENSIZ);
X            pstmp.s_dp = pdtmp;
X            pstmp.s_fp = pftmp;
X            for (fp = pftmp; fp < &pftmp[PFIELDS]; fp++)
X                fp->f_disp += s;
X            screen(&pstmp);
X        }
X    }
X}
SHAR_EOF
```

```
chmod 0664 patch.c || echo "restore of patch.c fails"
echo "x - extracting setuptmp.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > setuptmp.c &&
X#include "mt.h"
X#include "mtscrn.h"
X
Xdata_t setupdata[] = {
X    {  1, 32, "D110 RHYTHM SETUP" },
X    {  3, 10, "Tone (1-64 memory, 65-127 Rhythm, 128 off)" },
X    {  3, 57, "[   ]" },
X    {  5, 10, "Output Level (0-100)" },
X    {  5, 57, "[   ]" },
X    {  7, 10, "Panpot (-7 to 7) Right to Left" },
X    {  7, 58, "[  ]" },
X    {  9, 10, "Reverb Switch (0-off, 1-on)" },
X    {  9, 59, "[ ]" }
X};
X
Xfield_t setupfield[] = {
X/*   y   x   type   len   max   offset   disp */
X{    3, 58,     0,    3,  128,       1,     0 },
X{    5, 58,     0,    3,  100,       0,     1 },
X{    7, 59,     0,    2,   14,      -7,     2 },
X{    9, 60,     0,    1,    1,       0,     3 }
X};
X
Xuchar *mtgetdata();
Xint mtputdata();
Xint mtkeys();
Xint mtdone();
X
Xscreen_t setupscreen = {
X    setupdata,
X    sizeof(setupdata)/sizeof(data_t),
X    setupfield,
X    sizeof(setupfield)/sizeof(field_t),
X    mtgetdata,
X    mtputdata,
X    mtkeys,
X    mtdone
X};
X
Xssetuptmp()
X{
X    screen(&setupscreen);
X}
SHAR_EOF
chmod 0664 setuptmp.c || echo "restore of setuptmp.c fails"
echo "x - extracting src.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > src.c &&
X#include <osbind.h>
X#include <stdio.h>
X#include "mt.h"
```

```
X#include "mttype.h"
X
Xuchar midibuf[270];    /* the midi buffer */
Xuchar mtwork[256];     /* the workspace area */
X
Xtype_t *ctp;       /* current type pointer */
Xint centry;        /* current entry */
Xint memflg;        /* memory flag */
Xuchar outfile[15];     /* the destination file name */
X
Xlong c3tol();
X
Xuchar mtget[] = {
X    0xf0,          /* start excl */
X    0x41,          /* Roland Id */
X    0x10,          /* device 16 */
X    0x16,          /* mt32 model id */
X    0x11,          /* read command */
X    0x00,          /* address byte 2 */
X    0x00,          /* address byte 1 */
X    0x00,          /* address byte 0 */
X    0x00,          /* size byte 2 */
X    0x00,          /* size byte 1 */
X    0x00,          /* size byte 0 */
X    0x00,          /* checksum */
X    0xf7      /* excl end */
X};
X
X#define MTADDR         5
X#define MTSIZ          8
X#define MTDATA         8
X#define MTGSUM         11
X#define MTCMD          4
X#define MTSEND         0x12
X
Xtype_t typetab[] = {
X    { T_TIMBRE,       8,   8,  { 0x05, 0x00, 0x00 }, 128, "tmb" },
X    { T_TIMBRETMP,   16,  16,  { 0x03, 0x00, 0x00 },   9, "tmt" },
X    { T_TONE,       256, 246,  { 0x08, 0x00, 0x00 },  64, "tnm" },
X    { T_TONETMP,    246, 246,  { 0x04, 0x00, 0x00 },   8, "tnt" },
X    { T_SYSTEM,      33,  33,  { 0x10, 0x00, 0x00 },   1, "sys" },
X    { T_SETUPTMP,     4,   4,  { 0x03, 0x01, 0x10 },  64, "set" },
X    { T_PATCH,      128, 128,  { 0x06, 0x00, 0x00 },  64, "ptm" }
X};
X
Xmtsrc(type, entryn)
X{
X    long addr;
X    register type_t *tp;
X    register uchar *cp;
X    int i;
X    int len;
X    int chksum;
```

```
X
X    tp = &typetab[type];
X    if (entryn < 0 || entryn > tp->t_max) {
X         printf("bad source entry number\n");
X         return(-1);
X    }
X    addr = c3tol(tp->t_base);
X    addr += entryn * tp->t_psiz;
X    ltoc3(addr, &mtget[MTADDR]);
X    ltoc3((long)tp->t_vsiz, &mtget[MTSIZ]);
X    midiflush();
X    chksum = 0;
X    for (cp = mtget + MTADDR; cp < &mtget[sizeof(mtget) - 2]; cp++)
X         chksum += *cp;
X    mtget[MTGSUM] = (0 - chksum)&0x7f;
X    Midiws(sizeof(mtget), mtget);
X    cp = midibuf;
X    *cp = 0;
X    for (addr = 0; addr < 100000l; addr++) {
X         if (Bconstat(MIDI)) {
X              *cp = Bconin(MIDI);
X              if ((cp - midibuf) >= 255)
X                   break;
X              if (*cp == 0xf7)
X                   break;
X              *(++cp) = 0;
X         }
X    }
X    if (*cp != 0xf7) {
X         printf("incomplete midi message received\n");
X         return(-1);
X    }
X    len = (cp - midibuf);
X    chksum = 0;
X    for (i = MTADDR; i < len; i++)
X         chksum += midibuf[i];
X    if (chksum&0x7f)
X         return(-1);
X    len -= MTDATA;
X    memcpy(midibuf + MTDATA, mtwork, len);
X    return(0);
X}
X
Xmtsrcf(type, filename)
Xuchar *filename;
X{
X    register type_t *tp;
X    FILE *fd;
X    int len;
X
X    tp = &typetab[type];
X    strcpy(outfile, filename);
X    bstrip(outfile);
```

```
X    strcat(outfile, ".");
X    strcat(outfile, tp->t_filesuf);  /* extension */
X    fd = fopen(outfile, "rb");
X    if (fd == 0) {
X         printf("open error on %s\n", outfile);
X         return(-1);
X    }
X    len = tp->t_vsiz + 10;
X    if (fread(midibuf, 1, len, fd) != len) {
X         printf("read len error on %s\n", outfile);
X         fclose(fd);
X         return(-1);
X    }
X    fclose(fd);
X    len -= MTDATA;
X    memcpy(midibuf + MTDATA, mtwork, len);
X    return(0);
X}
X
Xmtdst(type, entryn)
X{
X    register type_t *tp;
X
X    tp = &typetab[type];
X    ctp = tp;
X    if (entryn >= tp->t_max) {
X         printf("bad dst entry number\n");
X         return(-1);
X    }
X    centry = entryn;
X    memflg = 1;
X    return(0);
X}
X
Xmtdstf(type, entryn, filename)
Xuchar *filename;
X{
X    register type_t *tp;
X
X    tp = &typetab[type];
X    ctp = tp;
X    centry = entryn;
X    memflg = 0;
X    strcpy(outfile, filename);
X    bstrip(outfile);
X    strcat(outfile, ".");
X    strcat(outfile, tp->t_filesuf);
X    return(0);
X}
X
Xlong
Xc3tol(cp)
Xuchar *cp;
```

```
X{
X    register long l;
X
X    l = 0;
X    l |= (*cp++)&0x7f;
X    l <<= 7;
X    l |= (*cp++)&0x7f;
X    l <<= 7;
X    l |= *cp&0x7f;
X    return(l);
X}
X
Xltoc3(l, cp)
Xregister long l;
Xregister uchar *cp;
X{
X    cp += 2;
X    *cp-- = l&0x7f;
X    l >>= 7;
X    *cp-- = l&0x7f;
X    l >>= 7;
X    *cp = l&0x7f;
X}
X
Xmidiflush()
X{
X    while (Bconstat(MIDI))
X         Bconin(MIDI);
X}
X
Xuchar *
Xmtgetdata(disp)
X{
X    register uchar *cp;
X
X    cp = mtwork + disp;
X    return(cp);
X}
X
Xmtputdata()
X{
X    int len;
X
X    if (memflg == 0)
X         return;
X    len = mtformat();  /* sticks msg into midibuf */
X    Midiws(len, midibuf);
X}
X
Xmtdone()
X{
X    int len;
X    FILE *fd;
```

```
X
X    if (memflg)
X         return;
X
X    len = mtformat();  /* sticks msg into midibuf */
X    fd = fopen(outfile, "wb");
X    if (fd == 0) {
X         kclear();
X         printf("open error on output file %s\n", outfile);
X         sleep(2);
X         return;
X    }
X    if (fwrite(midibuf, 1, len, fd) != len) {
X         kclear();
X         printf("write error on %s\n", outfile);
X         fclose(fd);
X         sleep(2);
X         return;
X    }
X    fclose(fd);
X    return;
X}
X
X/*
X * format an entry into midibuf for output or write to file
X */
Xmtformat()
X{
X    register type_t *tp;
X    register uchar *cp;
X    long l;
X    int chksum;
X    int len;
X
X    tp = ctp;
X    memcpy(mtget, midibuf, MTCMD);
X    midibuf[MTCMD] = MTSEND;
X    l = c3tol(tp->t_base) + (centry*tp->t_psiz);
X    ltoc3(l, midibuf + MTADDR);
X    memcpy(mtwork, midibuf + MTDATA, tp->t_vsiz);
X    chksum = 0;
X    for (cp = midibuf + MTADDR; cp < (midibuf + tp->t_vsiz + MTDATA); cp++)
X         chksum += *cp;
X    *cp++ = (0 - chksum)&0x7f;
X    *cp++ = 0xf7;
X    len = tp->t_vsiz + 10;
X    return(len);
X}
X
X/*
X * clobber everything after the first blank by changing
X * the first blank to a zero.  Blanks are ok for the roland
X * but bad for filenames
```

```
X */
Xbstrip(cp)
Xregister char *cp;
X{
X    char *index();
X    register char *cp1;
X
X    cp1 = index(cp, ' ');
X    if (cp1)
X         *cp1 = 0;
X}
SHAR_EOF
chmod 0664 src.c || echo "restore of src.c fails"
echo "x - extracting stdio.h (Text)"
sed 's/^X//' << 'SHAR_EOF' > stdio.h &&
X/*
X * Standard I/O library for Atari ST.
X */
X
X#ifndef STDIO_H
X#define STDIO_H  STDIO_H
Xtypedef struct    FILE {
X    unsigned char *_cp,         /* current character ptr */
X             *_dp,         /* ptr to start of data in buffer */
X             *_bp;         /* buffer pointer */
X    int  _cc;          /* character count */
X    int  (*_gt)(),     /* getc function */
X         (*_pt)();     /* putc function */
X    char _ff;          /* flags; see below */
X    char _fd;          /* file descriptor (reqd by reopen) */
X    int  _uc;          /* ungot char */
X}   FILE;
X#endif
X
X#define NULL ((char *)0)
X#define EOF  (-1)
X#define BUFSIZ   (1<<9)
X#define _NFILE   20
X
Xextern   FILE _stdin, _stdout, _stderr, *_fp[_NFILE];
X
X/* Flags in _ff */
X#define _FINUSE  01
X#define _FSTBUF  02       /* setbuf was called */
X#define _FUNGOT  04       /* ungotten char present */
X#define _FASCII  010      /* ascii mode (\r\n instead of \n) */
X#define _FEOF    0100
X#define _FERR    0200
X
X#define _ep(fp)       ((fp)->_bp+BUFSIZ)
X
Xchar    *gets();
Xchar    *fgets();
```

```
XFILE    *fopen();
XFILE    *freopen();
XFILE    *fdopen();
XFILE    *_stropen();
Xchar    *malloc();
Xchar    *sbrk();
Xlong    ftell();
Xvoid    puts();
Xvoid    fputs();
Xvoid    setbuf();
Xlong    bdos();
X
X#define getc(fp)  fgetc(fp)
X#define putc(c, fp)   fputc(c, fp)
X#define  getchar() getc(stdin)
X#define  putchar(c)    putc(c, stdout)
X#define  feof(fp)  ((fp)->_ff&(_FEOF|_FERR))
X#define  ferror(fp)    ((fp)->_ff&_FERR)
X#define  clearerr(fp)  ((fp)->_ff &= ~(_FERR|_FEOF))
X#define  fileno(fp)    ((fp)->_fd)
X#define  wdleng()  (16)
X
X#define  stdin    (&_stdin)
X#define  stdout   (&_stdout)
X#define  stderr   (&_stderr)
SHAR_EOF
chmod 0664 stdio.h || echo "restore of stdio.h fails"
echo "x - extracting system.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > system.c &&
X#include "mt.h"
X#include "mtscrn.h"
X
Xuchar *mtgetdata();
Xint mtputdata();
Xint mtkeys();
Xint mtdone();
X
Xdata_t sysdata[] = {
X    {  1, 28, "MT32 System Area" },
X    {  3, 20, "Master Tune   [   ]  0 is 440, .2 hz increments" },
X    {  5, 20, "Patch Name    [          ]" },
X{  7,  1, "Reverb Mode Room 0-1, Hall 2-3, Plate 4, Tap delay 5-7, off 8  [ ]
(0-8)" },
X    {  9,  8, "Reverb Time  [ ]  1-8" },
X    {  9, 34, "Reverb Level  [ ]  0-7" },
X    { 11, 20, "Partial Reserves (0-32), total of 32" },
X    { 13,  0, "ch 1 [  ] ch 2 [  ] ch 3 [  ] ch 4 [  ] ch 5 [  ]" },
X    { 13, 50, "ch 6 [  ] ch 7 [  ] ch 8 [  ]" },
X    { 14,  0, "Rhythm [  ]" },
X    { 15, 16, "Midi Channel For Each Part (1-16, 17 = OFF)" },
X    { 17,  0, "ch 1 [  ] ch 2 [  ] ch 3 [  ] ch 4 [  ] ch 5 [  ]" },
X    { 17, 50, "ch 6 [  ] ch 7 [  ] ch 8 [  ]" },
X    { 18,  0, "Rhythm [  ]" }
```

```
X};
X
Xfield_t sysfields[] = {
X/* y    x  type  len  max  offset  disp */
X{  3,  35,    0,   3, 127,    -64,    0 },    /* master tune */
X{  5,  35,    1,  10,   0,      0,   23 },    /* patch name */
X{  7,  65,    0,   1,   8,      0,    1 },      /* reverb mode */
X{  9,  22,    0,   1,   7,      1,    2 },      /* reverb time */
X{  9,  49,    0,   1,   7,      0,    3 },    /* reverb level */
X{  13,  6,    0,   2,  32,      0,    4 },    /* reserve ch 1 */
X{  13, 16,    0,   2,  32,      0,    5 },    /* reserve ch 2 */
X{  13, 26,    0,   2,  32,      0,    6 },    /* reserve ch 3 */
X{  13, 36,    0,   2,  32,      0,    7 },    /* reserve ch 4 */
X{  13, 46,    0,   2,  32,      0,    8 },    /* reserve ch 5 */
X{  13, 56,    0,   2,  32,      0,    9 },    /* reserve ch 6 */
X{  13, 66,    0,   2,  32,      0,   10 },    /* reserve ch 7 */
X{  13, 76,    0,   2,  32,      0,   11 },    /* reserve ch 8 */
X{  14,  8,    0,   2,  32,      0,   12 },    /* reserve rhythm */
X{  17,  6,    0,   2,  16,      1,   13 },      /* midi channel 1 */
X{  17, 16,    0,   2,  16,      1,   14 },      /* midi channel 2 */
X{  17, 26,    0,   2,  16,      1,   15 },      /* midi channel 3 */
X{  17, 36,    0,   2,  16,      1,   16 },      /* midi channel 4 */
X{  17, 46,    0,   2,  16,      1,   17 },      /* midi channel 5 */
X{  17, 56,    0,   2,  16,      1,   18 },      /* midi channel 6 */
X{  17, 66,    0,   2,  16,      1,   19 },      /* midi channel 6 */
X{  17, 76,    0,   2,  16,      1,   20 },      /* midi channel 7 */
X{  18,  8,    0,   2,  16,      1,   21 }       /* midi channel R */
X};
X
Xscreen_t sysscreen = {
X    sysdata,
X    sizeof(sysdata)/sizeof(data_t),
X    sysfields,
X    sizeof(sysfields)/sizeof(field_t),
X    mtgetdata,
X    mtputdata,
X    mtkeys,
X    mtdone
X};
X
Xssystem()
X{
X    screen(&sysscreen);
X}
SHAR_EOF
chmod 0664 system.c || echo "restore of system.c fails"
echo "x - extracting timbre.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > timbre.c &&
X#include "mt.h"
X#include "mtscrn.h"
X
Xuchar *mtgetdata();
Xint mtputdata();
```

```
Xint mtkeys();
Xint mtdone();
X
Xdata_t timbredata[] = {
X    {  1, 26, "D110 TIMBRE MEMORY" },
X    {  5,  8, "Tone Group (Group A, Group B, i/c Memory, Rhythm)" },
X    {  5, 58, "[ ]    (1-4)" },
X    {  7,  8, "Tone Number" },
X    {  7, 58, "[  ]   (1-64)" },
X    {  9,  8, "Key Shift" },
X    {  9, 58, "[   ]  (-24 to 24)" },
X    { 11,  8, "Fine Tune" },
X    { 11, 58, "[   ]  (-50 to 50)" },
X    { 13,  8, "Bender Range (half steps)" },
X    { 13, 58, "[  ]   (0 to 24)" },
X    { 15,  8, "Assign Mode (single-last,s-first,multi-last,m-f)" },
X    { 15, 58, "[ ]    (1-4)" },
X    { 17,  8, "Output assign (-1 no reverb, 0 reverb, 1 thru 6" },
X    { 17, 58, "[ ]    (-1-6)" }
X};
X
Xfield_t timbrefield[] = {
X/*  x   y   type   len   max   offset   disp */
X{   5, 59,     0,    1,    3,       1,     0 },        /* timbre group */
X{   7, 59,     0,    2,   63,       1,     1 },        /* timbre number */
X{   9, 59,     0,    3,   48,     -24,     2 },        /* Key Shift */
X{  11, 59,     0,    3,  100,     -50,     3 },        /* Fine Tune */
X{  13, 59,     0,    2,   24,       0,     4 },        /* Bender Range */
X{  15, 59,     0,    1,    3,       1,     5 },        /* Assign Mode */
X{  17, 59,     0,    1,    7,      -1,     6 }         /* Output assign */
X};
X
Xscreen_t timbrescrn[] = {
X    timbredata,
X    sizeof(timbredata)/sizeof(data_t),
X    timbrefield,
X    sizeof(timbrefield)/sizeof(field_t),
X    mtgetdata,
X    mtputdata,
X    mtkeys,
X    mtdone
X};
X
Xstimbre()
X{
X    screen(timbrescrn);
X}
SHAR_EOF
chmod 0664 timbre.c || echo "restore of timbre.c fails"
echo "x - extracting timbretm.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > timbretm.c &&
X#include "mt.h"
X#include "mtscrn.h"
```

```
X
Xuchar *mtgetdata();
Xint mtputdata();
Xint mtkeys();
Xint mtdone();
X
Xdata_t timbretdata[] = {
X    {  1, 26, "D110 TIMBRE TMP MEMORY" },
X    {  5,  8, "Tone Group (Group A, Group B, Memory, Rhythm)" },
X    {  5, 58, "[ ]    (1-4)" },
X    {  7,  8, "Tone Number" },
X    {  7, 58, "[  ]   (1-64)" },
X    {  9,  8, "Key Shift" },
X    {  9, 58, "[   ]  (-24 to 24)" },
X    { 11,  8, "Fine Tune" },
X    { 11, 58, "[   ]  (-50 to 50)" },
X    { 13,  8, "Bender Range (half steps)" },
X    { 13, 58, "[  ]   (0 to 24)" },
X    { 15,  8, "Assign Mode (single-last,s-first,multi-last,m-f)" },
X    { 15, 58, "[ ]    (1-4)" },
X    { 17,  8, "Output assign (-1 no reverb, 0 reverb, 1 thru 6)" },
X    { 17, 58, "[ ]    (-1-6)" },
X    { 19,  8, "Output level" },
X    { 19, 58, "[   ]  (0-100)" },
X    { 21,  8, "Panpot (Right to Left)" },
X    { 21, 58, "[   ]  (-7 to 7)" },
X    { 23,  8, "[   ]  lower kbd range" },
X    { 23, 40, "[   ]  upper kbd range" }
X};
X
Xfield_t timbretfield[] = {
X/*  x   y   type   len   max   offset   disp */
X{   5, 59,     0,    1,    3,       1,     0 },        /* timbre group */
X{   7, 59,     0,    2,   63,       1,     1 },        /* timbre number */
X{   9, 59,     0,    3,   48,     -24,     2 },        /* Key Shift */
X{  11, 59,     0,    3,  100,     -50,     3 },        /* Fine Tune */
X{  13, 59,     0,    2,   24,       0,     4 },        /* Bender Range */
X{  15, 59,     0,    1,    3,       1,     5 },        /* Assign Mode */
X{  17, 59,     0,    1,    7,      -1,     6 },        /* Output assign */
X{  19, 59,     0,    3,  100,       0,     8 },        /* output level */
X{  21, 59,     0,    2,   14,      -7,     9 },        /* panpot */
X{  23,  9,     0,    3,  127,       0,    10 },        /* lower kbd */
X{  23, 41,     0,    3,  127,       0,    11 }         /* upper kbd */
X};
X
Xscreen_t timbretscrn[] = {
X    timbretdata,
X    sizeof(timbretdata)/sizeof(data_t),
X    timbretfield,
X    sizeof(timbretfield)/sizeof(field_t),
X    mtgetdata,
X    mtputdata,
X    mtkeys,
```

```
X    mtdone
X};
X
Xstimbretmp()
X{
X    screen(timbretscrn);
X}
SHAR_EOF
chmod 0664 timbretm.c || echo "restore of timbretm.c fails"
echo "x - extracting tone.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > tone.c &&
X#include "mt.h"
X#include "mtscrn.h"
X
X#define TDATASIZ  (sizeof(tonedata))
X#define TDATAS        (TDATASIZ/sizeof(data_t))
X#define TFIELDSIZ (sizeof(tonefield))
X#define TFIELDS       (TFIELDSIZ/sizeof(field_t))
X#define TSCREENSIZ    (sizeof(tonescreen))
X
Xuchar *mtgetdata();
Xint mtputdata();
Xint mtkeys();
Xint mtdone();
X
Xextern uchar mtwork[];
X
Xdata_t tonedata[] = {
X{  0,  0, "WAVE GENERATOR:" },
X{  0, 26, "Pitch Coarse (0-96)   [  ]  Pitch Fine (-50-50) [   ]" },
X{  1,  0, "Keyfollow (-3-12)  [   ]" },
X{  1, 26, "Pitch Bender enable    [ ]  Waveform 1-SQ 2-SAW   [ ]" },
X{  2,  0, "PCM Sample # 1-128 [   ]" },
X{  2, 26, "Pulse Width (0-100)  [   ]  WG Velo Sense (-7-7) [  ]" },
X{  4,  0, "PITCH ENVELOPE:" },
X{  4, 26, "Depth  (0-10)         [  ]  Velo Sense (0-100)  [   ]" },
X{  5,  0, "Time Keyfollow (0-4) [ ]" },
X{  5, 26, "Time 1 (0-100)       [   ]  Time 2 (0-100)      [   ]" },
X{  6,  0, "Time 3 (0-100)     [   ]" },
X{  6, 26, "Time 4 (0-100)       [   ]  Level 0 (-50-50)    [   ]" },
X{  7,  0, "Level 1 (-50-50)   [   ]" },
X{  7, 26, "Level 2 (-50-50)     [   ]  Sustain (-50-50)    [   ]" },
X{  8,  0, "End Level (-50-50) [   ]" },
X{  9,  0, "P-LFO Rate (0-100) [   ]" },
X{  9, 26, "P-LFO Depth (0-100)  [   ]  P-LFO Mod Sens 0-100[   ]" },
X{ 11,  0, "TIME VARIABLE FILTER:" },
X{ 11, 26, "Cutoff Freq (0-100)  [   ]  Resonance (0-30)     [  ]" },
X{ 12,  0, "Keyfollow (-3-12)   [  ]" },
X{ 12, 26, "Bias Point (-64-64)  [   ]  Bias Level (-7-7)    [  ]" },
X{ 13,  0, "Env Depth (0-100)  [   ]" },
X{ 13, 26, "Velo Sens (0-100)    [   ]  Depth Keyfollow 0-4   [ ]" },
X{ 14,  0, "Time Keyfollow 0-4   [ ]" },
X{ 14, 26, "Time 1 (0-100)       [   ]  Time 2 (0-100)      [   ]" },
```

```
X{ 15,  0, "Time 3 (0-100)     [   ]" },
X{ 15, 26, "Time 4 (0-100)       [   ]  Time 5 (0-100)      [   ]" },
X{ 16,  0, "Level 1 (0-100)    [   ]" },
X{ 16, 26, "Level 2 (0-100)      [   ]  Level 3 (0-100)     [   ]" },
X{ 17,  0, "Sustain (0-100)    [   ]" },
X{ 19,  0, "TIME VARIABLE AMPLIFIER:" },
X{ 19, 26, "Level (0-100)        [   ]  Velo Sens (0-100)   [   ]" },
X{ 20,  0, "Bias 1 Point -64-64[   ]" },
X{ 20, 26, "Bias 1 Level (-12-0) [   ]  Bias 2 Point -64-64 [   ]" },
X{ 21,  0, "Bias 2 Level -12-0 [   ]" },
X{ 21, 26, "Time Keyfollow (0-4)   [ ]  Time Velo Follow 0-4  [ ]" },
X{ 22,  0, "Time 1 (0-100)     [   ]" },
X{ 22, 26, "Time 2 (0-100)       [   ]  Time 3 (0-100)      [   ]" },
X{ 23,  0, "Time 4 (0-100)     [   ]" },
X{ 23, 26, "Time 5 (0-100)       [   ]  Level 1 (0-100)     [   ]" },
X{ 24,  0, "Level2 (0-100)     [   ]" },
X{ 24, 26, "Level 3 (0-100)      [   ]  Sustain (0-100)     [   ]" }
X};
X
Xfield_t tonefield[] = {
X/*   y   x   type   len   max   offset   disp */
X{    0, 49,     0,    2,   96,       0,     0 },  /* pitch coarse */
X{    0, 75,     0,    3,  200,     -50,     1 },  /* pitch fine */
X{    1, 20,     0,    3,   16,      -3,     2 },  /* keyfollow */
X{    1, 50,     0,    1,    1,       0,     3 },  /* Bender enable */
X{    1, 77,     0,    1,    1,       1,     4 },  /* Waveform */
X{    2, 20,     0,    3,  127,       1,     5 },  /* PCM # */
X{    2, 48,     0,    3,  100,       0,     6 },  /* Pulse Width */
X{    2, 76,     0,    2,   14,      -7,     7 },  /* WG velo sens */
X/* P-ENVELOPE Generator */
X{    4, 49,     0,    2,   10,       0,     8 },  /* P-env Depth */
X{    4, 75,     0,    3,  100,       0,     9 },  /* Velo Sens */
X{    5, 22,     0,    1,    4,       0,  0x0a },  /* time keyfollow */
X{    5, 48,     0,    3,  100,       0,  0x0b },  /* Time 1 */
X{    5, 75,     0,    3,  100,       0,  0x0c },  /* time 2 */
X{    6, 20,     0,    3,  100,       0,  0x0d },  /* time 3 */
X{    6, 48,     0,    3,  100,       0,  0x0e },  /* time 4 */
X{    6, 75,     0,    3,  100,     -50,  0x0f },  /* level 0 */
X{    7, 20,     0,    3,  100,     -50,  0x10 },  /* level 1 */
X{    7, 48,     0,    3,  100,     -50,  0x11 },  /* level 2 */
X{    7, 75,     0,    3,  100,     -50,  0x12 },  /* sustain */
X{    8, 20,     0,    3,  100,     -50,  0x13 },  /* End Level */
X{    9, 20,     0,    3,  100,       0,  0x14 },  /* P-LFO Rate */
X{    9, 48,     0,    3,  100,       0,  0x15 },  /* P-LFO Depth */
X{    9, 75,     0,    3,  100,       0,  0x16 },  /* P-LFO Mod sens */
X/* Time Variable Filter */
X{   11, 48,     0,    3,  100,       0,  0x17 },  /* Cutoff Freq */
X{   11, 76,     0,    2,   30,       0,  0x18 },        /* Resonance */
X{   12, 21,     0,    2,   16,      -3,  0x19 },  /* Keyfollow */
X{   12, 48,     0,    3,  127,     -64,  0x1a },  /* bias point */
X{   12, 76,     0,    2,   14,      -7,  0x1b },  /* bias level */
X{   13, 20,     0,    3,  100,       0,  0x1c },  /* envelope depth */
X{   13, 48,     0,    3,  100,       0,  0x1d },  /* Velo Sens */
```

```
X{   13, 77,     0,    1,    4,       0,  0x1e },   /* Depth keyf */
X{   14, 22,     0,    1,    4,       0,  0x1f },   /* Time keyf */
X{   14, 48,     0,    3,  100,       0,  0x20 },   /* Time 1 */
X{   14, 75,     0,    3,  100,       0,  0x21 },   /* Time 2 */
X{   15, 20,     0,    3,  100,       0,  0x22 },   /* Time 3 */
X{   15, 48,     0,    3,  100,       0,  0x23 },   /* Time 4 */
X{   15, 75,     0,    3,  100,       0,  0x24 },   /* Time 5 */
X{   16, 20,     0,    3,  100,       0,  0x25 },   /* Level 1 */
X{   16, 48,     0,    3,  100,       0,  0x26 },   /* Level 2 */
X{   16, 75,     0,    3,  100,       0,  0x27 },   /* Level 3 */
X{   17, 20,     0,    3,  100,       0,  0x28 },   /* Sustain */
X/* Time Variable Amplifier */
X{   19, 48,     0,    3,  100,       0,  0x29 },   /* Main level */
X{   19, 75,     0,    3,  100,       0,  0x2a },   /* Velo Sens */
X{   20, 20,     0,    3,  127,     -64,  0x2b },   /* Bias 1 point */
X{   20, 48,     0,    3,   12,     -12,  0x2c },   /* Bias 1 level */
X{   20, 75,     0,    3,  127,     -64,  0x2d },   /* Bias 2 point */
X{   21, 20,     0,    3,   12,     -12,  0x2e },   /* Bias 2 level */
X{   21, 50,     0,    1,    4,       0,  0x2f },   /* Time keyfollow */
X{   21, 77,     0,    1,    4,       0,  0x30 },   /* Time velo keyf */
X{   22, 20,     0,    3,  100,       0,  0x31 },   /* Time 1 */
X{   22, 48,     0,    3,  100,       0,  0x32 },   /* Time 2 */
X{   22, 75,     0,    3,  100,       0,  0x33 },   /* Time 3 */
X{   23, 20,     0,    3,  100,       0,  0x34 },   /* Time 4 */
X{   23, 48,     0,    3,  100,       0,  0x35 },   /* Time 5 */
X{   23, 75,     0,    3,  100,       0,  0x36 },   /* Level 1 */
X{   24, 20,     0,    3,  100,       0,  0x37 },   /* Level 2 */
X{   24, 48,     0,    3,  100,       0,  0x38 },   /* Level 3 */
X{   24, 75,     0,    3,  100,       0,  0x39 }        /* Sustain */
X};
X
Xscreen_t tonescreen = {
X    tonedata,
X    sizeof(tonedata)/sizeof(data_t),
X    tonefield,
X    sizeof(tonefield)/sizeof(field_t),
X    mtgetdata,
X    mtputdata,
X    mtkeys,
X    mtdone
X};
X
Xdata_t tdtmp[TDATAS];
Xfield_t tftmp[TFIELDS];
Xscreen_t tstmp;
X
Xdata_t commondata[] = {
X    {  1, 30, "D110 Tone Common Area" },
X    {  4, 20, "Tone Name   [" },
X    {  4, 43, "]" },
X    {  6, 20, "Structure Partials 1 and 2 (1-13)       [  ]" },
X    {  8, 20, "Structure Partials 3 and 4 (1-13)       [  ]" },
X    { 10, 20, "Partial Mute (T4, T3, T2, T1)           [    ]" },
```

```
X    { 12, 20, "Envelope Mode (0-normal, 1-no sustain)  [ ]" },
X    { 14, 20, "Partial to edit (0-none, 1-4)           [ ]" }
X};
X
Xfield_t commonfield[] = {
X/*   y   x   type   len   max   offset   disp */
X{    4, 33,     1,   10,    0,       0,     0 },  /* tone name */
X{    6, 61,     0,    2,   12,       1,    10 },  /* structure 1&2 */
X{    8, 61,     0,    2,   12,       1,    11 },  /* structure 3&4 */
X{   10, 61,     2,    4,   16,       0,    12 },  /* Mute */
X{   12, 61,     0,    1,    1,       0,    13 },  /* envelope mode */
X{   14, 61,     0,    1,    4,       0,    250 }  /* partial to edit */
X};
X
Xscreen_t commonscreen = {
X    commondata,
X    sizeof(commondata)/sizeof(data_t),
X    commonfield,
X    sizeof(commonfield)/sizeof(field_t),
X    mtgetdata,
X    mtputdata,
X    mtkeys,
X    mtdone
X};
X
Xstone()
X{
X    int s;
X    register field_t *fp;
X
X    mtwork[250] = 0;       /* partial to edit */
X    for (;;) {
X         if (screen(&commonscreen) < 0)
X              return;
X         switch (s = mtwork[250]) {
X         case 0:            /* no partial edits */
X              return;
X         case 1:
X         case 2:
X         case 3:
X         case 4:
X              s--;
X              s *= 58;      /* timbre data area */
X              s += 0x0e;         /* common data size */
X              memcpy(tonedata, tdtmp, TDATASIZ);
X              memcpy(tonefield, tftmp, TFIELDSIZ);
X              memcpy(&tonescreen, &tstmp, TSCREENSIZ);
X              tstmp.s_dp = tdtmp;
X              tstmp.s_fp = tftmp;
X              for (fp = tftmp; fp < &tftmp[TFIELDS]; fp++)
X                   fp->f_disp += s;
X              screen(&tstmp);
X         }
```

```
X    }
X}
SHAR_EOF
chmod 0664 tone.c || echo "restore of tone.c fails"
echo "x - extracting top.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > top.c &&
X#include <osbind.h>
X#include "mt.h"
X#include "mtscrn.h"
X#include "mttype.h"
X
Xuchar srctype;
Xuchar dsttype;
Xuchar srcentry;
Xuchar dstentry;
Xuchar srcfile[9];
Xuchar dstfile[9];
Xuchar edittype;
Xuchar midichan = 1;
X
Xdata_t topdata[] = {
X    { 1, 26, "D110 Patch Editor" },
X    { 5, 12, "Src Type   [ ]  Entry #  [   ]  Src File  [" },
X    { 5, 63, "]" },
X    { 7, 12, "Dst Type   [ ]  Entry #  [   ]  Dst File  [" },
X    { 7, 63, "]" },
X    { 9, 16, "Type  1 - File" },
X    { 9, 40, "Midi Test Channel (1-16) [  ]" },
X    { 10, 22, "2 - D110 Memory" },
X    { 12, 12, "Edit Type  [ ]    1 - Timbre     (*.tmb)" },
X    { 13, 30, "2 - Timbre Tmp (*.tmt)" },
X    { 14, 30, "3 - Tone       (*.tnm)" },
X    { 15, 30, "4 - Tone Tmp   (*.tnt)" },
X    { 16, 30, "5 - System     (*.sys)" },
X    { 17, 30, "6 - Setup Tmp  (*.set)" },
X    { 18, 30, "7 - Patch      (*.ptm)" },
X    { 19, 30, "8 - Exit" }
X};
X
Xfield_t topfield[] = {
X    { 5, 24, 0, 1, 1, 1, 0 },
X    { 5, 38, 0, 3, 127, 1, 1 },
X    { 5, 55, 1, 8, 0, 0, 2 },
X    { 7, 24, 0, 1, 1, 1, 3 },
X    { 7, 38, 0, 3, 127, 1, 4 },
X    { 7, 55, 1, 8, 0, 0, 5 },
X    { 9, 66, 0, 2, 16, 1, 7 },
X    { 12, 24, 0, 1, 6, 1, 6 }
X};
X
Xuchar *topget();
Xint topput();
Xint mtkeys();
```

```
X
Xscreen_t topscrn = {
X    topdata,
X    sizeof(topdata)/sizeof(data_t),
X    topfield,
X    sizeof(topfield)/sizeof(field_t),
X    topget,
X    topput,
X    mtkeys,
X    topput
X};
X
Xtop()
X{
X    int ret;
X
X    edittype = 7;
X    for (;;) {
X         if (screen(&topscrn) < 0)
X              break;
X         if (edittype == 7)
X              break;
X         ksetcrsr(0, 20);
X         if (srctype == 0)  /* file */
X              ret = mtsrcf(edittype, srcfile);
X         else
X              ret = mtsrc(edittype, srcentry);
X         if (ret == -1) {
X              sleep(1);
X              continue;
X         }
X         if (dsttype == 0)  /* file */
X              ret = mtdstf(edittype, dstentry, dstfile);
X         else
X              ret = mtdst(edittype, dstentry);
X         if (ret == -1) {
X              sleep(1);
X              continue;
X         }
X         switch (edittype) {
X         case T_TIMBRE:              /* timbre */
X              stimbre();
X              break;
X         case T_TIMBRETMP:      /* timbre tmp */
X              stimbretmp();
X              break;
X         case T_TONE:           /* tone */
X         case T_TONETMP:             /* tone tmp */
X              stone();
X              break;
X         case T_SYSTEM:              /* system */
X              ssystem();
X              break;
```

```
X         case T_SETUPTMP:       /* setup tmp */
X              ssetuptmp();
X              break;
X         case T_PATCH:          /* patch memory */
X              spatch();
X              break;
X         }
X    }
X    ksetcrsr(0, 23);
X    exit(0);
X}
X
Xuchar *
Xtopget(disp)
X{
X    switch (disp) {
X    case 0:
X         return(&srctype);
X         break;
X    case 1:
X         return(&srcentry);
X         break;
X    case 2:
X         return(srcfile);
X         break;
X    case 3:
X         return(&dsttype);
X         break;
X    case 4:
X         return(&dstentry);
X         break;
X    case 5:
X         return(dstfile);
X         break;
X    case 6:
X         return(&edittype);
X         break;
X    case 7:
X         return(&midichan);
X         break;
X    default:
X         printf("error\n");
X         exit(0);
X    }
X}
X
Xtopput()
X{
X}
X
SHAR_EOF
chmod 0664 top.c || echo "restore of top.c fails"
echo "x - extracting makefile (Text)"
```

```
sed 's/^X//' << 'SHAR_EOF' > makefile &&
XSRC =   makefile \
X    mt.h \
X    mtscrn.h \
X    mt.c \
X    mtscrn.c \
X    top.c \
X    src.c \
X    timbrre.c \
X    timbretm.c \
X    tone.c \
X    system.c \
X    setuptmp.c \
X    mttype.h \
X    keys.c \
X    patch.c
X
XOBJS =  mt.o \
X    mtscrn.o \
X    top.o \
X    src.o \
X    timbre.o \
X    timbretm.o \
X    tone.o \
X    system.o \
X    setuptmp.o \
X    keys.o \
X    patch.o
X
Xd110.tos: $(OBJS)
X    cc -o d110.tos $(OBJS) -lar
X
Xcp:
X    cp $(SRC) a:\src\d110
SHAR_EOF
chmod 0664 makefile || echo "restore of makefile fails"
echo "x - extracting d110.doc (Text)"
sed 's/^X//' << 'SHAR_EOF' > d110.doc &&
XDoc for d110 patch editor:
X
XThe program is full screen, the editing keys are:
X
Xnumeric fields:
X    F1  -16
X    F2  -8
X    F3  -4
X    F4  -2
X    F5  -1
X    F6  +1
X    F7  +2
X    F8  +4
X    F9  +8
X    F10 +10
```

```
X
XAlphanumeric fields:
X    There is no backspace (sorry about the bug)
X    a-z, A-Z, and 0-9 are permitted.
X
XNext and previous fields are the up and down arrow keys.
XLeft and right arrow keys allow moving around inside
Xan alphanumeric field.
X
XWhen the output is directed to the d110 the
Xpatch editor will edit in live mode:  every change
Xwill be reflected immediately.  Messages from a keyboard
Xare copied from the ST's MIDI in and sent the MIDI out
Xwhich allows a keyboard to be used to play the synth
Xwhile making changes to the patch.  Be warned that the d110
Xwill reject patch changes while sounding a tone.  So all
Xnotes on the synth keyboard should be off when making changes
Xon the ST's keyboard.
X
XWhen the output is directed to the d110 the alphabetic keys on
Xthe ST's keyboard form all of the notes in the C arpeggio for
Xall 8 octaves.  The space bar sends an all notes off msg.
XWhile more privitive than having a full keyboard,
Xit allows for live editing without a keyboard.
X
XCabling is difficult, there is no setup that lets you source
Xand sink to the D110 and have a synth keyboard hooked up
Xall at the same time.  You get one or the other but not
Xboth.  To do live editing you need the D110's out connected
Xto the ST's in and the ST's out connected to the D110's in.
X
XFor synth keyboard usage, use the above cabling method first and
Xcopy the patches to disk files.  Then switch cables so that the
Xsynth kbd is connected to the ST's in.  The D110's out will
Xbe disconnected.  When done editing the new patch will be in the
XD110's memory.  Another cable switch and you can copy the new patch
Xto a disk file.
X
XWhat all the patch fields are:
XAll of my names closely track Roland's names.
XIf you have doubts, check the MIDI implementation reference
Xin the back since that is where my names came from.
X
XFile formats:
XMy file formats are exactly the sysex message it would take to
Xsend the data to the d110.  Thus, if your sequencer allows importing
Xsuch files, you can store song setup information in your seqence.
XThe d110 needs about a half a second to digest each such message.
XI did not include a program that would dump sysex files to the
XMIDI port.  Such a program is trivial (for programmers) and I might
Xdo it for those interested.
X
X             - kurt
```

```
X
XOn compiling the source:
X    the memcpy.s file is not referenced in the makefile.
X    It is the only item needed from the archive libar.a
X    (-lar).
SHAR_EOF
chmod 0664 d110.doc || echo "restore of d110.doc fails"
exit 0


From -7454995309406552728
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,9632ecc35c150b8e,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-01-18 21:55:10 PST
Path: utzoo!attcan!uunet!lll-winken!ncis.llnl.gov!ncis!helios.ee.lbl.gov!pasteur!
ucbvax!agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i091: mt32edit -- Patch editor for MT32 synthesizer
Keywords: shar, C
Message-ID: <6023@saturn.ucsc.edu>
Date: 19 Jan 89 05:55:10 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1792
Approved: koreth@ssyx.ucsc.edu

Submitted-by: kurth@sco.com (Kurt Hutchinson)
Posting-number: Volume 1, Issue 91
Archive-name: mt32edit

#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      keys.c
#      memcpy.s
#      mt.c
#      mt.h
#      mtscrn.c
#      mtscrn.h
#      mttype.h
#      osbind.h
#      patch.c
#      patchtmp.c
#      setuptmp.c
#      src.c
#      stdio.h
#      system.c
#      timbre.c
#      top.c
#      makefile
#      mt32.doc
```

```
#
echo "x - extracting keys.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > keys.c &&
X#include <osbind.h>
X#include "mt.h"
X
X/*
X * keyboard to midi translator
X */
X
Xunsigned char midichan;
X
X/* note on, note #, note intensity */
Xunsigned char mtnote[3] = { 0x92, 0x00, 0x7d };
Xunsigned char mtnoteoff[3] = { 0xb2, 0x7b, 0x00 };
X
Xmtkeys(l)
Xlong l;
X{
X    int k;
X    int in;
X
X    k = l>>16;
X    if (k == 0x62) {
X         help();
X         return;
X    }
X    mtnote[1] = 0;
X    in = 0;
X    k = l&0x7f;
X    switch (k) {
X    case '?':
X         help();
X         return;
X    case '1':
X         in = 0x07;
X         break;
X    case '2':
X         in = 0x17;
X         break;
X    case '3':
X         in = 0x37;
X         break;
X    case '4':
X         in = 0x47;
X         break;
X    case '5':
X         in = 0x57;
X         break;
X    case '6':
X         in = 0x67;
X         break;
X    case '7':
```

```
X         in = 0x77;
X         break;
X    case '8':
X         in = 0x7f;
X         break;
X    case 'q':
X         mtnote[1] = 24;         /* C */
X         break;
X    case 'w':
X         mtnote[1] = 28;         /* E */
X         break;
X    case 'e':
X         mtnote[1] = 31;         /* G */
X         break;
X    case 'r':
X         mtnote[1] = 36;         /* C */
X         break;
X    case 't':
X         mtnote[1] = 40;         /* E */
X         break;
X    case 'y':
X         mtnote[1] = 43;         /* G */
X         break;
X    case 'u':
X         mtnote[1] = 48;         /* C */
X         break;
X    case 'i':
X         mtnote[1] = 52;         /* E */
X         break;
X    case 'o':
X         mtnote[1] = 55;         /* G */
X         break;
X    case 'p':
X         mtnote[1] = 60;         /* C */
X         break;
X    case '[':
X         mtnote[1] = 64;         /* E */
X         break;
X    case ']':
X         mtnote[1] = 67;         /* G */
X         break;
X    case 'a':
X         mtnote[1] = 72;         /* C */
X         break;
X    case 's':
X         mtnote[1] = 76;         /* E */
X         break;
X    case 'd':
X         mtnote[1] = 79;         /* G */
X         break;
X    case 'f':
X         mtnote[1] = 84;         /* C */
X         break;
```

```
X    case 'g':
X         mtnote[1] = 88;        /* E */
X         break;
X    case 'h':
X         mtnote[1] = 91;        /* G */
X         break;
X    case 'j':
X         mtnote[1] = 96;        /* C */
X         break;
X    case 'k':
X         mtnote[1] = 100;   /* E */
X         break;
X    case 'l':
X         mtnote[1] = 103;   /* G */
X         break;
X    case ';':
X         mtnote[1] = 108;   /* C */
X         break;
X    case ' ':
X         mtnoteoff[0] &= 0xf0;
X         mtnoteoff[0] |= midichan;
X         Midiws(3, mtnoteoff);
X         return;
X    }
X    if (in)
X         mtnote[2] = in;
X    if (mtnote[1]) {
X         mtnote[0] &= 0xf0;
X         mtnote[0] |= midichan;
X         Midiws(3, mtnote);
X    }
X}
X
Xhelp()
X{
X    kclear();
Xprintf("F1 to F10 are -16, -8, -4, -2, -1, +1, +2, +4, +8, +16 respectively\n");
Xprintf("Arrow keys select field\n");
Xprintf("Escape goes to previous level\n");
Xprintf("Enter takes selected action or previous level\n");
Xprintf("\nMidi test Bed only works on numeric fields\n");
Xprintf("keys 1 to 8 select intensity\n");
Xprintf("keys q-] and a-; are all notes in the C arpeggio\n");
X    printf("\n\tenter any key to continue\n");
X    while (Bconstat(CON));
X         Bconin(CON);
X    drawscrn();
X}
SHAR_EOF
chmod 0444 keys.c || echo "restore of keys.c fails"
echo "x - extracting memcpy.s (Text)"
sed 's/^X//' << 'SHAR_EOF' > memcpy.s &&
X    .shri
```

```
X    .globl   memcpy_
Xmemcpy_:
X    link a6, $0
X    movea.l  8(a6), a0
X    movea.l  12(a6), a1
X    clr.l    d0
X    move.w   16(a6), d0
X    cmpi.l   $15, d0
X    bgt  long
Xtop:
X    subq.l   $1, d0
Xloop:
X    move.b   (a0)+, (a1)+
X    dbf  d0, loop
Xdone:
X    unlk a6
X    rts
X
Xlong:
X    move.l   a0, d1
X    move.l   a1, d2
X    btst $0, d1
X    beq  z1
X    btst $0, d2
X    beq  loop
X    move.b   (a0)+, (a1)+  / both odd
X    subq.l   $1, d0
Xaligned:
X    move.l   d0, d1
X    lsr.l    $2, d1        / long word count
X    subq.l   $1, d1
Xlloop:
X    move.l   (a0)+, (a1)+
X    dbf  d1, lloop
X    andi.l   $3, d0
X    beq  done
X    jmp  top
X
Xz1:
X    btst $0, d2
X    beq  aligned
X    jmp  loop
X SHAR_EOF
chmod 0444 memcpy.s || echo "restore of memcpy.s fails"
echo "x - extracting mt.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > mt.c &&
Xmain()
X{
X    top();
X}
SHAR_EOF
chmod 0444 mt.c || echo "restore of mt.c fails"
echo "x - extracting mt.h (Text)"
```

```
sed 's/^X//' << 'SHAR_EOF' > mt.h &&
X#define CON  2
X#define MIDI 3
X#define RQMAX     16400
X
X#define uchar     unsigned char
X#define uint unsigned int
SHAR_EOF
chmod 0444 mt.h || echo "restore of mt.h fails"
echo "x - extracting mtscrn.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > mtscrn.c &&
X#include <osbind.h>
X#include "mt.h"
X#include "mtscrn.h"
X
Xscreen_t *csp;         /* current screen */
Xint cfld;         /* current field index */
Xuchar sbuf[81];        /* ascii field data buffer */
Xint cindex;       /* current index into an ascii field */
X
Xscreen(sp)
Xregister screen_t *sp;
X{
X    csp = sp;
X    drawscrn();
X
X    scrinput();
X}
X
Xscrinput()
X{
X    register screen_t *sp;
X    register field_t *fp;
X    uchar *cp;
X    register int c;
X    long l;
X
X    midiflush();
X    setfld(csp->s_fp);
X    for (;;) {
X         while (Bconstat(CON) == 0) { /* midi pass thru feature */
X              if (Bconstat(MIDI)) {
X                   c = Bconin(MIDI);
X                   Bconout(MIDI, c);
X              }
X         }
X         l = Bconin(CON);   /* we have a character to read */
X         if ((l&0xff) == 0x1b) {/* escape is no write */
X              return(-1);
X         }
X         sp = csp;
X         fp = &sp->s_fp[cfld];
X         c = (l>>16)&0xff;
```

```
X         switch (c) {
X         case 0x1c:    /* enter done with screen */
X              (*sp->s_done)();
X              return(0);
X         case 0x4b:    /* left arrow */
X              if (cfld)
X                   fp--;
X              setfld(fp);
X              continue;
X         case 0x48:    /* up arrow */
X              c = fp->f_y;
X              while (cfld) {
X                   fp--;
X                   cfld--;
X                   if (fp->f_y < c)
X                        break;
X              }
X              while (cfld) {
X                   if ((fp-1)->f_y == fp->f_y) {
X                        fp--;
X                        cfld--;
X                   }
X                   else
X                        break;
X              }
X              setfld(fp);
X              continue;
X         case 0x4d:    /* right arrow */
X              if ((cfld + 1) < sp->s_fieldsiz)
X                   fp++;
X              setfld(fp);
X              continue;
X         case 0x50:    /* down arrow */
X              c = fp->f_y;
X              while ((cfld + 1) < sp->s_fieldsiz) {
X                   fp++;
X                   cfld++;
X                   if (fp->f_y > c)
X                        break;
X              }
X              setfld(fp);
X              continue;
X         }
X         if (fp->f_type == 1) { /* ascii field */
X              c = l&0x7f;
X              if ((c >= 'a' && c <= 'z') ||
X                   (c >= 'A' && c <= 'Z') ||
X                   (c >= '0' && c <= '9') ||
X                   (c == '.') || (c == ' ')) {
X                   cp = (*sp->s_get)(fp->f_disp);
X                   *(cp + cindex) = c;
X                   if ((cindex + 1) < fp->f_len)
X                        cindex++;
```

```
X                    (*sp->s_put)(fp->f_disp);
X               }
X               else {
X                    (*sp->s_ukn)(l);
X                    continue;
X               }
X          }
X          else {              /* numeric field */
X               switch (c) {
X               case 0x3b:     /* F1 */
X                    fchange(-16);
X                    break;
X               case 0x3c:     /* F2 */
X                    fchange(-8);
X                    break;
X               case 0x3d:     /* F3 */
X                    fchange(-4);
X                    break;
X               case 0x3e:     /* F4 */
X                    fchange(-2);
X                    break;
X               case 0x3f:     /* F5 */
X                    fchange(-1);
X                    break;
X               case 0x40:     /* F6 */
X                    fchange(1);
X                    break;
X               case 0x41:     /* F7 */
X                    fchange(2);
X                    break;
X               case 0x42:     /* F8 */
X                    fchange(4);
X                    break;
X               case 0x43:     /* F9 */
X                    fchange(8);
X                    break;
X               case 0x44:     /* F10 */
X                    fchange(16);
X                    break;
X               default:
X                    (*sp->s_ukn)(l);
X                    continue;
X               }
X          }
X          drawfield(fp);
X          ksetcrsr(fp->f_x + cindex, fp->f_y);
X     }
X}
X
Xfchange(change)
X{
X     register screen_t *sp;
X     register field_t *fp;
```

```
X    register uchar *cp;
X    int c;
X
X    sp = csp;
X    fp = &sp->s_fp[cfld];
X    cp = (*sp->s_get)(fp->f_disp);
X    c = *cp + change;
X    if (c < 0)
X         c = 0;
X    if (c > fp->f_max)
X         c = fp->f_max;
X    *cp = c;
X    (*sp->s_put)(fp->f_disp);
X}
X
Xsetfld(fp)
Xregister field_t *fp;
X{
X    cindex = 0;
X    cfld = fp - csp->s_fp;
X    ksetcrsr(fp->f_x, fp->f_y);
X}
X
Xdrawscrn()
X{
X    kclear();
X    drawdata();
X    drawflds();
X}
X
Xdrawdata()
X{
X    register screen_t *sp;
X    register data_t *dp;
X    register int i;
X
X    sp = csp;
X    dp = sp->s_dp;
X    for (i = 0; i < sp->s_datasiz; i++) {
X         ksetcrsr(dp->d_x, dp->d_y);
X         kouts(dp->d_cp);
X         dp++;
X    }
X}
X
Xdrawflds()
X{
X    register screen_t *sp;
X    register field_t *fp;
X    register int i;
X
X    sp = csp;
X    fp = sp->s_fp;
```

```
X    for (i = 0; i < sp->s_fieldsiz; i++) {
X         drawfield(fp);
X         fp++;
X    }
X}
X
Xdrawfield(fp)
Xregister field_t *fp;
X{
X    register screen_t *sp;
X    uchar *cp;
X    int t;
X    int i;
X
X    sp = csp;
X    ksetcrsr(fp->f_x, fp->f_y);
X    switch (fp->f_type) {
X    case 2:            /* binary */
X         cp = (*sp->s_get)(fp->f_disp);
X         t = *cp + fp->f_offset;
X         cp = sbuf + fp->f_len - 1;
X         for (i = 0; i < fp->f_len; i++) {
X              if ((t & 1) == 0)
X                   *cp-- = '0';
X              else
X                   *cp-- = '1';
X              t >>= 1;
X         }
X         sbuf[fp->f_len] = 0;
X         break;
X    case 1:            /* ascii */
X         cp = (*sp->s_get)(fp->f_disp);
X         memcpy(cp, sbuf, fp->f_len);
X         sbuf[fp->f_len] = 0;
X         pad(sbuf, fp->f_len);
X         break;
X    case 0:            /* decimal */
X         cp = (*sp->s_get)(fp->f_disp);
X         t = *cp + fp->f_offset;
X         sprintf(sbuf, "%d", t);
X         sbuf[fp->f_len] = 0;
X         pad(sbuf, fp->f_len);
X    }
X    kouts(sbuf);
X}
X
Xkclear()
X{
X    Bconout(CON, 0x1b);
X    Bconout(CON, 'E');
X}
X
Xksetcrsr(x, y)
```

```
X{
X    Bconout(CON, 0x1b);
X    Bconout(CON, 'Y');
X    Bconout(CON, y + 0x20);
X    Bconout(CON, x + 0x20);
X}
X
Xkoutc(c)
X{
X    Bconout(CON, c);
X}
X
Xkouts(cp)
Xregister uchar *cp;
X{
X    while (*cp)
X         Bconout(CON, *cp++);
X}
X
Xpad(cp, len)
Xregister uchar *cp;
X{
X    int i;
X
X    cp[len] = 0;
X    i = strlen(cp);
X    while (i < len)
X         cp[i++] = ' ';
X}
SHAR_EOF
chmod 0444 mtscrn.c || echo "restore of mtscrn.c fails"
echo "x - extracting mtscrn.h (Text)"
sed 's/^X//' << 'SHAR_EOF' > mtscrn.h &&
X/*
X * defines for the two screen structures
X */
X
Xstruct datadef {
X    uchar    d_y;      /* x coordinate for string */
X    uchar    d_x;      /* y coordinate for string */
X    uchar    *d_cp;         /* the string itself */
X};
X
Xtypedef struct datadef data_t;
X
Xstruct fielddef {
X    uchar    f_y;      /* x coordinate of field */
X    uchar    f_x;      /* y coordinate of field */
X    uchar    f_type;        /* 0 is number, non-zero is ascii */
X    uchar    f_len;         /* field length */
X    uchar    f_max;         /* maximum value */
X    char f_offset; /* offset to be added before display */
X    int  f_disp;        /* displacement of data */
```

```
X};
X
Xtypedef struct fielddef field_t;
X
Xstruct screendef {
X    data_t   *s_dp;        /* data pointer */
X    int  s_datasiz;    /* size of data array */
X    field_t *s_fp;         /* field pointer */
X    int  s_fieldsiz;   /* size of field array */
X    uchar    *(*s_get)();  /* get data function */
X    int  (*s_put)();   /* put data function */
X    int  (*s_ukn)();   /* unknown character function */
X    int  (*s_done)();  /* end of screen routine */
X};
X
Xtypedef struct screendef screen_t;
SHAR_EOF
chmod 0444 mtscrn.h || echo "restore of mtscrn.h fails"
echo "x - extracting mttype.h (Text)"
sed 's/^X//' << 'SHAR_EOF' > mttype.h &&
X/*
X * defines for the source type structure
X */
X
Xstruct mtypedef {
X    int  t_type;       /* source type */
X    int  t_psiz;       /* physical size (displacement between adjacent
X                      entries */
X    int  t_vsiz;       /* virtual size (size of read and write
X                      requests */
X    uchar    t_base[3];    /* base memory address in mt32 */
X    int  t_max;        /* max entries in mt32 memory area */
X    uchar    t_filesuf[6]; /* file suffix for this type */
X};
X
Xtypedef struct mtypedef type_t;
X
X#define T_PATCH       0    /* patch */
X#define T_PATCHTMP    1    /* patch tmp */
X#define T_TIMBRE  2    /* timbre memory */
X#define T_TIMBRETMP   3    /* timbre tmp memory */
X#define T_SYSTEM  4    /* system memory area */
X#define T_SETUPTMP    5    /* setup tmp area */
SHAR_EOF
chmod 0444 mttype.h || echo "restore of mttype.h fails"
echo "x - extracting osbind.h (Text)"
sed 's/^X//' << 'SHAR_EOF' > osbind.h &&
X/**
X*
X* This file contains macro definitions for use with the Atari specific
X* functions gemdos,bios and xbios
X*
X**/
```

```
X
Xextern long bios();
Xextern long xbios();
Xextern long gemdos();
X
X/* GEMDOS functions (trap #1) */
X
X#define Pterm0()      gemdos(0x0)
X#define Cconin()      gemdos(0x1)
X#define Cconout(a)         gemdos(0x2,a)
X#define Cauxin()      gemdos(0x3)
X#define Cauxout(a)         gemdos(0x4,a)
X#define Cprnout(a)         gemdos(0x5,a)
X#define Crawio(a)     gemdos(0x6,a)
X#define Crawcin()     gemdos(0x7)
X#define Cnecin()      gemdos(0x8)
X#define Cconws(a)     gemdos(0x9,a)
X#define Cconrs(a)     gemdos(0x0a,a)
X#define Cconis()      gemdos(0x0b)
X#define Dsetdrv(a)         gemdos(0x0e,a)
X#define Cconos()      gemdos(0x10)
X#define Cprnos()      gemdos(0x11)
X#define Cauxis()      gemdos(0x12)
X#define Cauxos()      gemdos(0x13)
X#define Dgetdrv()     gemdos(0x19)
X#define Fsetdta(a)         gemdos(0x1a,a)
X#define Super(a)      gemdos(0x20,a)
X#define Tgetdate()         gemdos(0x2a)
X#define Tsetdate(a)        gemdos(0x2b,a)
X#define Tgettime()         gemdos(0x2c)
X#define Tsettime(a)        gemdos(0x2d,a)
X#define Fgetdta()     gemdos(0x2f)
X#define Sversion()         gemdos(0x30)
X#define Ptermres(a,b)      gemdos(0x31,a,b)
X#define Dfree(a,b)         gemdos(0x36,a,b)
X#define Dcreate(a)         gemdos(0x39,a)
X#define Ddelete(a)         gemdos(0x3a,a)
X#define Dsetpath(a)        gemdos(0x3b,a)
X#define Fcreate(a,b)       gemdos(0x3c,a,b)
X#define Fopen(a,b)         gemdos(0x3d,a,b)
X#define Fclose(a)     gemdos(0x3e,a)
X#define Fread(a,b,c)       gemdos(0x3f,a,b,c)
X#define Fwrite(a,b,c)      gemdos(0x40,a,b,c)
X#define Fdelete(a)         gemdos(0x41,a)
X#define Fseek(a,b,c)       gemdos(0x42,a,b,c)
X#define Fattrib(a,b,c)     gemdos(0x43,a,b,c)
X#define Fdup(a)            gemdos(0x45,a)
X#define Fforce(a,b)        gemdos(0x46,a,b)
X#define Dgetpath(a,b)      gemdos(0x47,a,b)
X#define Malloc(a)     gemdos(0x48,a)
X#define Mfree(a)      gemdos(0x49,a)
X#define Mshrink(a,b)       gemdos(0x4a,0,a,b)   /* NOTE: Null parameter added */
X#define Pexec(a,b,c,d)     gemdos(0x4b,a,b,c,d)
```

```
X#define Pterm(a)       gemdos(0x4c,a)
X#define Fsfirst(a,b)       gemdos(0x4e,a,b)
X#define Fsnext()       gemdos(0x4f)
X#define Frename(a,b,c)     gemdos(0x56,a,b,c)
X#define Fdatime(a,b,c)     gemdos(0x57,a,b,c)
X
X/* BIOS functions (trap #13) */
X
X#define  Getmpb(a)     bios(0,a)
X#define Bconstat(a)        bios(1,a)
X#define Bconin(a)      bios(2,a)
X#define Bconout(a,b)       bios(3,a,b)
X#define Rwabs(a,b,c,d,e)   bios(4,a,b,c,d,e)
X#define Setexc(a,b)        bios(5,a,b)
X#define  Tickcal()     bios(6)
X#define  Getbpb(a)     bios(7,a)
X#define Bcostat(a)         bios(8,a)
X#define Mediach(a)         bios(9,a)
X#define Drvmap()       bios(10)
X#define Getshift(a)        bios(11,a)
X
X/* XBIOS functions (trap #14) */
X
X#define Initmous(a,b,c)         (void) xbios(0,a,b,c)
X#define Physbase()              (char *) xbios(2)
X#define Logbase()               (char *) xbios(3)
X#define Getrez()                (int) xbios(4)
X#define Setscreen(a,b,c)        (void) xbios(5,a,b,c)
X#define Setpallete(a)           (void) xbios(6,a)
X#define Setcolor(a,b)           (int) xbios(7,a,b)
X#define Floprd(a,b,c,d,e,f,g)   (int) xbios(8,a,b,c,d,e,f,g)
X#define Flopwr(a,b,c,d,e,f,g)   (int) xbios(9,a,b,c,d,e,f,g)
X#define  Flopfmt(a,b,c,d,e,f,g,h,i) (int) xbios(10,a,b,c,d,e,f,g,h,i)
X/* XBIOS function 11 is unused */
X#define Midiws(a,b)        (void) xbios(12,a,b)
X#define Mfpint(a,b)        (void) xbios(13,a,b)
X#define Iorec(a)       (char *) xbios(14,a)
X#define Rsconf(a,b,c,d,e,f)     (void) xbios(15,a,b,c,d,e,f)
X#define Keytbl(a,b,c)      (char *) xbios(16,a,b,c)
X#define Random()       xbios(17)
X#define Protobt(a,b,c,d)   (void) xbios(18,a,b,c,d)
X#define Flopver(a,b,c,d,e,f,g)  (int) xbios(19,a,b,c,d,e,f,g)
X#define Scrdmp()       (void) xbios(20)
X#define Cursconf(a,b)      (int) xbios(21,a,b)
X#define Settime(a)         (void) xbios(22,a)
X#define Gettime()      xbios(23)
X#define Bioskeys()         (void) xbios(24)
X#define Ikbdws(a,b)        (void) xbios(25,a,b)
X#define Jdisint(a)         (void) xbios(26,a)
X#define Jenabint(a)        (void) xbios(27,a)
X#define Giaccess(a,b)      (char) xbios(28,a,b)
X#define Offgibit(a)        (void) xbios(29,a)
X#define Ongibit(a)         (void) xbios(30,a)
```

```
X#define Xbtimer(a,b,c,d)   (void) xbios(31,a,b,c,d)
X#define Dosound(a)         (void) xbios(32,a)
X#define Setprt(a)     (int) xbios(33,a)
X#define Kbdvbase()         (char **) xbios(34)
X#define Kbrate(a,b)        (int) xbios(35,a,b)
X#define Prtblk(a)     (int) xbios(36,a)
X#define  Vsync()           (void) xbios(37)
X#define  Supexec(a)        (unsigned long) xbios(38,a)
X#define  Puntaes()     (void) xbios(39)
SHAR_EOF
chmod 0444 osbind.h || echo "restore of osbind.h fails"
echo "x - extracting patch.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > patch.c &&
X#include "mt.h"
X#include "mtscrn.h"
X
Xuchar *mtgetdata();
Xint mtputdata();
Xint mtkeys();
Xint mtdone();
X
Xdata_t patchdata[] = {
X    {  1, 26, "MT32 PATCH MEMORY" },
X    {  5,  8, "Timbre Group (Group A, Group B, Memory, Rhythm)" },
X    {  5, 58, "[ ]    (1-4)" },
X    {  7,  8, "Timbre Number" },
X    {  7, 58, "[  ]   (1-64)" },
X    {  9,  8, "Key Shift" },
X    {  9, 58, "[   ]  (-24 to 24)" },
X    { 11,  8, "Fine Tune" },
X    { 11, 58, "[   ]  (-50 to 50)" },
X    { 13,  8, "Bender Range (half steps)" },
X    { 13, 58, "[  ]   (0 to 24)" },
X    { 15,  8, "Assign Mode (single-last,s-first,multi-last,m-f)" },
X    { 15, 58, "[ ]    (1-4)" },
X    { 17,  8, "Reverb Switch (off, on)" },
X    { 17, 58, "[ ]    (0-1)" }
X};
X
Xfield_t patchfield[] = {
X/*  x   y   type   len   max   offset   disp */
X{   5, 59,     0,    1,    3,       1,     0 },        /* timbre group */
X{   7, 59,     0,    2,   63,       1,     1 },        /* timbre number */
X{   9, 59,     0,    3,   48,     -24,     2 },        /* Key Shift */
X{  11, 59,     0,    3,  100,     -50,     3 },        /* Fine Tune */
X{  13, 59,     0,    2,   24,       0,     4 },        /* Bender Range */
X{  15, 59,     0,    1,    3,       1,     5 },        /* Assign Mode */
X{  17, 59,     0,    1,    1,       0,     6 }         /* Reverb Switch */
X};
X
Xscreen_t patchscrn[] = {
X    patchdata,
X    sizeof(patchdata)/sizeof(data_t),
```

```
X    patchfield,
X    sizeof(patchfield)/sizeof(field_t),
X    mtgetdata,
X    mtputdata,
X    mtkeys,
X    mtdone
X};
X
Xspatch()
X{
X    screen(patchscrn);
X}
SHAR_EOF
chmod 0444 patch.c || echo "restore of patch.c fails"
echo "x - extracting patchtmp.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > patchtmp.c &&
X#include "mt.h"
X#include "mtscrn.h"
X
Xuchar *mtgetdata();
Xint mtputdata();
Xint mtkeys();
Xint mtdone();
X
Xdata_t patchtdata[] = {
X    {  1, 26, "MT32 PATCH MEMORY" },
X    {  5,  8, "Timbre Group (Group A, Group B, Memory, Rhythm)" },
X    {  5, 58, "[ ]    (1-4)" },
X    {  7,  8, "Timbre Number" },
X    {  7, 58, "[  ]   (1-64)" },
X    {  9,  8, "Key Shift" },
X    {  9, 58, "[   ]  (-24 to 24)" },
X    { 11,  8, "Fine Tune" },
X    { 11, 58, "[   ]  (-50 to 50)" },
X    { 13,  8, "Bender Range (half steps)" },
X    { 13, 58, "[  ]   (0 to 24)" },
X    { 15,  8, "Assign Mode (single-last,s-first,multi-last,m-f)" },
X    { 15, 58, "[ ]    (1-4)" },
X    { 17,  8, "Reverb Switch (off, on)" },
X    { 17, 58, "[ ]    (0-1)" },
X    { 19,  8, "Output level" },
X    { 19, 58, "[   ]  (0-100)" },
X    { 21,  8, "Panpot (Right to Left)" },
X    { 21, 58, "[   ]  (-7 to 7)" }
X};
X
Xfield_t patchtfield[] = {
X/*  x   y   type   len   max   offset   disp */
X{   5, 59,     0,    1,    3,       1,     0 },        /* timbre group */
X{   7, 59,     0,    2,   63,       1,     1 },        /* timbre number */
X{   9, 59,     0,    3,   48,     -24,     2 },        /* Key Shift */
X{  11, 59,     0,    3,  100,     -50,     3 },        /* Fine Tune */
X{  13, 59,     0,    2,   24,       0,     4 },        /* Bender Range */
```

```
X{  15, 59,     0,    1,    3,       1,     5 },        /* Assign Mode */
X{  17, 59,     0,    1,    1,       0,     6 },        /* Reverb Switch */
X{  19, 59,     0,    3,  100,       0,     8 },        /* output level */
X{  21, 59,     0,    2,   14,      -7,     9 }         /* panpot */
X};
X
Xscreen_t patchtscrn[] = {
X    patchtdata,
X    sizeof(patchtdata)/sizeof(data_t),
X    patchtfield,
X    sizeof(patchtfield)/sizeof(field_t),
X    mtgetdata,
X    mtputdata,
X    mtkeys,
X    mtdone
X};
X
Xspatchtmp()
X{
X    screen(patchtscrn);
X}
SHAR_EOF
chmod 0444 patchtmp.c || echo "restore of patchtmp.c fails"
echo "x - extracting setuptmp.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > setuptmp.c &&
X#include "mt.h"
X#include "mtscrn.h"
X
Xdata_t setupdata[] = {
X    {  1, 32, "MT32 RHYTHM SETUP" },
X    {  3, 10, "Timbre (1-64 memory, 65-94 Rhythm, 95 = off)" },
X    {  3, 58, "[  ]" },
X    {  5, 10, "Output Level (0-100)" },
X    {  5, 57, "[   ]" },
X    {  7, 10, "Panpot (-7 to 7) Right to Left" },
X    {  7, 58, "[  ]" },
X    {  9, 10, "Reverb Switch (0-off, 1-on)" },
X    {  9, 59, "[ ]" }
X};
X
Xfield_t setupfield[] = {
X/*   y   x   type   len   max   offset   disp */
X{    3, 59,     0,    2,   94,       1,     0 },
X{    5, 58,     0,    3,  100,       0,     1 },
X{    7, 59,     0,    2,   14,      -7,     2 },
X{    9, 60,     0,    1,    1,       0,     3 }
X};
X
Xuchar *mtgetdata();
Xint mtputdata();
Xint mtkeys();
Xint mtdone();
X
```

```
Xscreen_t setupscreen = {
X    setupdata,
X    sizeof(setupdata)/sizeof(data_t),
X    setupfield,
X    sizeof(setupfield)/sizeof(field_t),
X    mtgetdata,
X    mtputdata,
X    mtkeys,
X    mtdone
X};
X
Xssetuptmp()
X{
X    screen(&setupscreen);
X}
SHAR_EOF
chmod 0444 setuptmp.c || echo "restore of setuptmp.c fails"
echo "x - extracting src.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > src.c &&
X#include <osbind.h>
X#include <stdio.h>
X#include "mt.h"
X#include "mttype.h"
X
Xuchar midibuf[270];    /* the midi buffer */
Xuchar mtwork[256];     /* the workspace area */
X
Xtype_t *ctp;      /* current type pointer */
Xint centry;       /* current entry */
Xint memflg;       /* memory flag */
Xuchar outfile[15];     /* the destination file name */
X
Xlong c3tol();
X
Xuchar mtget[] = {
X    0xf0,         /* start excl */
X    0x41,         /* Roland Id */
X    0x10,         /* device 16 */
X    0x16,         /* mt32 model id */
X    0x11,         /* read command */
X    0x00,         /* address byte 2 */
X    0x00,         /* address byte 1 */
X    0x00,         /* address byte 0 */
X    0x00,         /* size byte 2 */
X    0x00,         /* size byte 1 */
X    0x00,         /* size byte 0 */
X    0x00,         /* checksum */
X    0xf7      /* excl end */
X};
X
X#define MTADDR        5
X#define MTSIZ         8
X#define MTDATA        8
```

```
X#define MTGSUM        11
X#define MTCMD         4
X#define MTSEND        0x12
X
Xtype_t typetab[] = {
X    { T_PATCH,       8,   8,  { 0x05, 0x00, 0x00 }, 128, "ptm" },
X    { T_PATCHTMP,   16,  16,  { 0x03, 0x00, 0x00 },   9, "ptt" },
X    { T_TIMBRE,    256, 246,  { 0x08, 0x00, 0x00 },  64, "tmb" },
X    { T_TIMBRETMP, 246, 246,  { 0x04, 0x00, 0x00 },   8, "tmt" },
X    { T_SYSTEM,     24,  24,  { 0x10, 0x00, 0x00 },   1, "sys" },
X    { T_SETUPTMP,    4,   4,  { 0x03, 0x01, 0x10 },  64, "set" }
X};
X
Xmtsrc(type, entryn)
X{
X    long addr;
X    register type_t *tp;
X    register uchar *cp;
X    int i;
X    int len;
X    int chksum;
X
X    tp = &typetab[type];
X    if (entryn < 0 || entryn > tp->t_max) {
X         printf("bad source entry number\n");
X         return(-1);
X    }
X    addr = c3tol(tp->t_base);
X    addr += entryn * tp->t_psiz;
X    ltoc3(addr, &mtget[MTADDR]);
X    ltoc3((long)tp->t_vsiz, &mtget[MTSIZ]);
X    midiflush();
X    chksum = 0;
X    for (cp = mtget + MTADDR; cp < &mtget[sizeof(mtget) - 2]; cp++)
X         chksum += *cp;
X    mtget[MTGSUM] = (0 - chksum)&0x7f;
X    Midiws(sizeof(mtget), mtget);
X    cp = midibuf;
X    *cp = 0;
X    for (addr = 0; addr < 100000l; addr++) {
X         if (Bconstat(MIDI)) {
X              *cp = Bconin(MIDI);
X              if ((cp - midibuf) >= 255)
X                   break;
X              if (*cp == 0xf7)
X                   break;
X              *(++cp) = 0;
X         }
X    }
X    if (*cp != 0xf7) {
X         printf("incomplete midi message received\n");
X         return(-1);
X    }
```

```
X    len = (cp - midibuf);
X    chksum = 0;
X    for (i = MTADDR; i < len; i++)
X         chksum += midibuf[i];
X    if (chksum&0x7f)
X         return(-1);
X    len -= MTDATA;
X    memcpy(midibuf + MTDATA, mtwork, len);
X    return(0);
X}
X
Xmtsrcf(type, filename)
Xuchar *filename;
X{
X    register type_t *tp;
X    FILE *fd;
X    int len;
X
X    tp = &typetab[type];
X    strcpy(outfile, filename);
X    bstrip(outfile);
X    strcat(outfile, ".");
X    strcat(outfile, tp->t_filesuf);  /* extension */
X    fd = fopen(outfile, "rb");
X    if (fd == 0) {
X         printf("open error on %s\n", outfile);
X         return(-1);
X    }
X    len = tp->t_vsiz + 10;
X    if (fread(midibuf, 1, len, fd) != len) {
X         printf("read len error on %s\n", outfile);
X         fclose(fd);
X         return(-1);
X    }
X    fclose(fd);
X    len -= MTDATA;
X    memcpy(midibuf + MTDATA, mtwork, len);
X    return(0);
X}
X
Xmtdst(type, entryn)
X{
X    register type_t *tp;
X
X    tp = &typetab[type];
X    ctp = tp;
X    if (entryn >= tp->t_max) {
X         printf("bad dst entry number\n");
X         return(-1);
X    }
X    centry = entryn;
X    memflg = 1;
X    return(0);
```

```
X}
X
Xmtdstf(type, entryn, filename)
Xuchar *filename;
X{
X    register type_t *tp;
X
X    tp = &typetab[type];
X    ctp = tp;
X    centry = entryn;
X    memflg = 0;
X    strcpy(outfile, filename);
X    bstrip(outfile);
X    strcat(outfile, ".");
X    strcat(outfile, tp->t_filesuf);
X    return(0);
X}
X
Xlong
Xc3tol(cp)
Xuchar *cp;
X{
X    register long l;
X
X    l = 0;
X    l |= (*cp++)&0x7f;
X    l <<= 7;
X    l |= (*cp++)&0x7f;
X    l <<= 7;
X    l |= *cp&0x7f;
X    return(l);
X}
X
Xltoc3(l, cp)
Xregister long l;
Xregister uchar *cp;
X{
X    cp += 2;
X    *cp-- = l&0x7f;
X    l >>= 7;
X    *cp-- = l&0x7f;
X    l >>= 7;
X    *cp = l&0x7f;
X}
X
Xmidiflush()
X{
X    while (Bconstat(MIDI))
X        Bconin(MIDI);
X}
X
Xuchar *
Xmtgetdata(disp)
```

```
X{
X    register uchar *cp;
X
X    cp = mtwork + disp;
X    return(cp);
X}
X
Xmtputdata()
X{
X    int len;
X
X    if (memflg == 0)
X         return;
X    len = mtformat();  /* sticks msg into midibuf */
X    Midiws(len, midibuf);
X}
X
Xmtdone()
X{
X    int len;
X    FILE *fd;
X
X    if (memflg)
X         return;
X
X    len = mtformat();  /* sticks msg into midibuf */
X    fd = fopen(outfile, "wb");
X    if (fd == 0) {
X         kclear();
X         printf("open error on output file %s\n", outfile);
X         sleep(2);
X         return;
X    }
X    if (fwrite(midibuf, 1, len, fd) != len) {
X         kclear();
X         printf("write error on %s\n", outfile);
X         fclose(fd);
X         sleep(2);
X         return;
X    }
X    fclose(fd);
X    return;
X}
X
X/*
X * format an entry into midibuf for output or write to file
X */
Xmtformat()
X{
X    register type_t *tp;
X    register uchar *cp;
X    long l;
X    int chksum;
```

```
X    int len;
X
X    tp = ctp;
X    memcpy(mtget, midibuf, MTCMD);
X    midibuf[MTCMD] = MTSEND;
X    l = c3tol(tp->t_base) + (centry*tp->t_psiz);
X    ltoc3(l, midibuf + MTADDR);
X    memcpy(mtwork, midibuf + MTDATA, tp->t_vsiz);
X    chksum = 0;
X    for (cp = midibuf + MTADDR; cp < (midibuf + tp->t_vsiz + MTDATA); cp++)
X         chksum += *cp;
X    *cp++ = (0 - chksum)&0x7f;
X    *cp++ = 0xf7;
X    len = tp->t_vsiz + 10;
X    return(len);
X}
X
X/*
X * clobber everything after the first blank by changing
X * the first blank to a zero.  Blanks are ok for the roland
X * but bad for filenames
X */
Xbstrip(cp)
Xregister char *cp;
X{
X    char *index();
X    register char *cp1;
X
X    cp1 = index(cp, ' ');
X    if (cp1)
X         *cp1 = 0;
X}
SHAR_EOF
chmod 0444 src.c || echo "restore of src.c fails"
echo "x - extracting stdio.h (Text)"
sed 's/^X//' << 'SHAR_EOF' > stdio.h &&
X/*
X * Standard I/O library for Atari ST.
X */
X
X#ifndef STDIO_H
X#define STDIO_H  STDIO_H
Xtypedef struct    FILE {
X    unsigned char *_cp,         /* current character ptr */
X              *_dp,         /* ptr to start of data in buffer */
X              *_bp;         /* buffer pointer */
X    int  _cc;          /* character count */
X    int  (*_gt)(),     /* getc function */
X         (*_pt)();     /* putc function */
X    char _ff;          /* flags; see below */
X    char _fd;          /* file descriptor (reqd by reopen) */
X    int  _uc;          /* ungot char */
X}   FILE;
```

```
X#endif
X
X#define  NULL ((char *)0)
X#define  EOF  (-1)
X#define  BUFSIZ    (1<<9)
X#define  _NFILE    20
X
Xextern   FILE _stdin, _stdout, _stderr, *_fp[_NFILE];
X
X/* Flags in _ff */
X#define  _FINUSE   01
X#define  _FSTBUF   02       /* setbuf was called */
X#define  _FUNGOT   04       /* ungotten char present */
X#define _FASCII    010      /* ascii mode (\r\n instead of \n) */
X#define  _FEOF     0100
X#define  _FERR     0200
X
X#define  _ep(fp)       ((fp)->_bp+BUFSIZ)
X
Xchar     *gets();
Xchar     *fgets();
XFILE     *fopen();
XFILE     *freopen();
XFILE     *fdopen();
XFILE     *_stropen();
Xchar     *malloc();
Xchar     *sbrk();
Xlong     ftell();
Xvoid     puts();
Xvoid     fputs();
Xvoid     setbuf();
Xlong     bdos();
X
X#define getc(fp)  fgetc(fp)
X#define putc(c, fp)    fputc(c, fp)
X#define  getchar() getc(stdin)
X#define  putchar(c)    putc(c, stdout)
X#define  feof(fp)  ((fp)->_ff&(_FEOF|_FERR))
X#define  ferror(fp)    ((fp)->_ff&_FERR)
X#define  clearerr(fp)  ((fp)->_ff &= ~(_FERR|_FEOF))
X#define  fileno(fp)    ((fp)->_fd)
X#define  wdleng()  (16)
X
X#define  stdin     (&_stdin)
X#define  stdout    (&_stdout)
X#define  stderr    (&_stderr)
SHAR_EOF
chmod 0444 stdio.h || echo "restore of stdio.h fails"
echo "x - extracting system.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > system.c &&
X#include "mt.h"
X#include "mtscrn.h"
X
```

```
Xuchar *mtgetdata();
Xint mtputdata();
Xint mtkeys();
Xint mtdone();
X
Xdata_t sysdata[] = {
X    {  1, 28, "MT32 System Area" },
X    {  3, 20, "Master Tune   [   ]  0 is 440, .2 hz increments" },
X    {  5, 20, "Master Volume [   ]  0 to 100" },
X    {  7,  8, "Reverb Mode (Room, Hall, Plate, Tap9 Delay)  [ ]  (0-3)" },
X    {  9,  8, "Reverb Time  [ ]  0-7" },
X    {  9, 34, "Reverb Level  [ ]  1-8" },
X    { 11, 20, "Partial Reserves (0-32), total of 32" },
X    { 13,  0, "ch 1 [  ] ch 2 [  ] ch 3 [  ] ch 4 [  ] ch 5 [  ]" },
X    { 13, 50, "ch 6 [  ] ch 7 [  ] ch 8 [  ]" },
X    { 14,  0, "Rhythm [  ]" },
X    { 15, 16, "Midi Channel For Each Part (1-16, 17 = OFF)" },
X    { 17,  0, "ch 1 [  ] ch 2 [  ] ch 3 [  ] ch 4 [  ] ch 5 [  ]" },
X    { 17, 50, "ch 6 [  ] ch 7 [  ] ch 8 [  ]" },
X    { 18,  0, "Rhythm [  ]" }
X};
X
Xfield_t sysfields[] = {
X/* y    x  type  len  max  offset  disp */
X{  3,  35,    0,   3, 127,    -64,    0 },    /* master tune */
X{  5,  35,    0,   3, 100,      0,   22 },    /* master volume */
X{  7,  54,    0,   1,   3,      0,    1 },      /* reverb mode */
X{  9,  22,    0,   1,   7,      0,    2 },      /* reverb time */
X{  9,  49,    0,   1,   7,      1,    3 },    /* reverb level */
X{  13,  6,    0,   2,  32,      0,    4 },    /* reserve ch 1 */
X{  13, 16,    0,   2,  32,      0,    5 },    /* reserve ch 2 */
X{  13, 26,    0,   2,  32,      0,    6 },    /* reserve ch 3 */
X{  13, 36,    0,   2,  32,      0,    7 },    /* reserve ch 4 */
X{  13, 46,    0,   2,  32,      0,    8 },    /* reserve ch 5 */
X{  13, 56,    0,   2,  32,      0,    9 },    /* reserve ch 6 */
X{  13, 66,    0,   2,  32,      0,   10 },    /* reserve ch 7 */
X{  13, 76,    0,   2,  32,      0,   11 },    /* reserve ch 8 */
X{  14,  8,    0,   2,  32,      0,   12 },    /* reserve rhythm */
X{  17,  6,    0,   2,  16,      1,   13 },      /* midi channel 1 */
X{  17, 16,    0,   2,  16,      1,   14 },      /* midi channel 2 */
X{  17, 26,    0,   2,  16,      1,   15 },      /* midi channel 3 */
X{  17, 36,    0,   2,  16,      1,   16 },      /* midi channel 4 */
X{  17, 46,    0,   2,  16,      1,   17 },      /* midi channel 5 */
X{  17, 56,    0,   2,  16,      1,   18 },      /* midi channel 6 */
X{  17, 66,    0,   2,  16,      1,   19 },      /* midi channel 6 */
X{  17, 76,    0,   2,  16,      1,   20 },      /* midi channel 7 */
X{  18,  8,    0,   2,  16,      1,   21 }       /* midi channel R */
X};
X
Xscreen_t sysscreen = {
X    sysdata,
X    sizeof(sysdata)/sizeof(data_t),
X    sysfields,
```

```
X    sizeof(sysfields)/sizeof(field_t),
X    mtgetdata,
X    mtputdata,
X    mtkeys,
X    mtdone
X};
X
Xssystem()
X{
X    screen(&sysscreen);
X}
SHAR_EOF
chmod 0444 system.c || echo "restore of system.c fails"
echo "x - extracting timbre.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > timbre.c &&
X#include "mt.h"
X#include "mtscrn.h"
X
X#define TDATASIZ  (sizeof(timbredata))
X#define TDATAS        (TDATASIZ/sizeof(data_t))
X#define TFIELDSIZ (sizeof(timbrefield))
X#define TFIELDS       (TFIELDSIZ/sizeof(field_t))
X#define TSCREENSIZ    (sizeof(timbrescreen))
X
Xuchar *mtgetdata();
Xint mtputdata();
Xint mtkeys();
Xint mtdone();
X
Xextern uchar mtwork[];
X
Xdata_t timbredata[] = {
X{  0,  0, "WAVE GENERATOR:" },
X{  0, 26, "Pitch Coarse (0-96)   [  ]  Pitch Fine (-50-50) [   ]" },
X{  1,  0, "Keyfollow (-3-12)  [   ]" },
X{  1, 26, "Pitch Bender enable    [ ]  Waveform 1-SQ 2-SAW   [ ]" },
X{  2,  0, "PCM Sample # 1-128 [   ]" },
X{  2, 26, "Pulse Width (0-100)  [   ]  WG Velo Sense (-7-7) [  ]" },
X{  4,  0, "PITCH ENVELOPE:" },
X{  4, 26, "Depth  (0-10)         [  ]  Velo Sense (0-100)  [   ]" },
X{  5,  0, "Time Keyfollow (0-4) [ ]" },
X{  5, 26, "Time 1 (0-100)       [   ]  Time 2 (0-100)      [   ]" },
X{  6,  0, "Time 3 (0-100)     [   ]" },
X{  6, 26, "Time 4 (0-100)       [   ]  Level 0 (-50-50)    [   ]" },
X{  7,  0, "Level 1 (-50-50)   [   ]" },
X{  7, 26, "Level 2 (-50-50)     [   ]  Sustain (-50-50)    [   ]" },
X{  8,  0, "End Level (-50-50) [   ]" },
X{  9,  0, "P-LFO Rate (0-100) [   ]" },
X{  9, 26, "P-LFO Depth (0-100)  [   ]  P-LFO Mod Sens 0-100[   ]" },
X{ 11,  0, "TIME VARIABLE FILTER:" },
X{ 11, 26, "Cutoff Freq (0-100)  [   ]  Resonance (0-30)     [  ]" },
X{ 12,  0, "Keyfollow (-3-12)   [  ]" },
X{ 12, 26, "Bias Point (-64-64)  [   ]  Bias Level (-7-7)    [  ]" },
```

```
X{ 13,  0, "Env Depth (0-100)  [   ]" },
X{ 13, 26, "Velo Sens (0-100)    [   ]  Depth Keyfollow 0-4   [ ]" },
X{ 14,  0, "Time Keyfollow 0-4   [ ]" },
X{ 14, 26, "Time 1 (0-100)       [   ]  Time 2 (0-100)      [   ]" },
X{ 15,  0, "Time 3 (0-100)     [   ]" },
X{ 15, 26, "Time 4 (0-100)       [   ]  Time 5 (0-100)      [   ]" },
X{ 16,  0, "Level 1 (0-100)    [   ]" },
X{ 16, 26, "Level 2 (0-100)      [   ]  Level 3 (0-100)     [   ]" },
X{ 17,  0, "Sustain (0-100)    [   ]" },
X{ 19,  0, "TIME VARIABLE AMPLIFIER:" },
X{ 19, 26, "Level (0-100)        [   ]  Velo Sens (0-100)   [   ]" },
X{ 20,  0, "Bias 1 Point -64-64[   ]" },
X{ 20, 26, "Bias 1 Level (-12-0) [   ]  Bias 2 Point -64-64 [   ]" },
X{ 21,  0, "Bias 2 Level -12-0 [   ]" },
X{ 21, 26, "Time Keyfollow (0-4)   [ ]  Time Velo Follow 0-4  [ ]" },
X{ 22,  0, "Time 1 (0-100)     [   ]" },
X{ 22, 26, "Time 2 (0-100)       [   ]  Time 3 (0-100)      [   ]" },
X{ 23,  0, "Time 4 (0-100)     [   ]" },
X{ 23, 26, "Time 5 (0-100)       [   ]  Level 1 (0-100)     [   ]" },
X{ 24,  0, "Level2 (0-100)     [   ]" },
X{ 24, 26, "Level 3 (0-100)      [   ]  Sustain (0-100)     [   ]" }
X};
X
Xfield_t timbrefield[] = {
X/*   y   x   type   len   max   offset   disp */
X{    0, 49,     0,    2,   96,       0,     0 },  /* pitch coarse */
X{    0, 75,     0,    3,  200,     -50,     1 },  /* pitch fine */
X{    1, 20,     0,    3,   16,      -3,     2 },  /* keyfollow */
X{    1, 50,     0,    1,    1,       0,     3 },  /* Bender enable */
X{    1, 77,     0,    1,    1,       1,     4 },  /* Waveform */
X{    2, 20,     0,    3,  127,       1,     5 },  /* PCM # */
X{    2, 48,     0,    3,  100,       0,     6 },  /* Pulse Width */
X{    2, 76,     0,    2,   14,      -7,     7 },  /* WG velo sens */
X/* P-ENVELOPE Generator */
X{    4, 49,     0,    2,   10,       0,     8 },  /* P-env Depth */
X{    4, 75,     0,    3,  100,       0,     9 },  /* Velo Sens */
X{    5, 22,     0,    1,    4,       0,  0x0a },  /* time keyfollow */
X{    5, 48,     0,    3,  100,       0,  0x0b },  /* Time 1 */
X{    5, 75,     0,    3,  100,       0,  0x0c },  /* time 2 */
X{    6, 20,     0,    3,  100,       0,  0x0d },  /* time 3 */
X{    6, 48,     0,    3,  100,       0,  0x0e },  /* time 4 */
X{    6, 75,     0,    3,  100,     -50,  0x0f },  /* level 0 */
X{    7, 20,     0,    3,  100,     -50,  0x10 },  /* level 1 */
X{    7, 48,     0,    3,  100,     -50,  0x11 },  /* level 2 */
X{    7, 75,     0,    3,  100,     -50,  0x12 },  /* sustain */
X{    8, 20,     0,    3,  100,     -50,  0x13 },  /* End Level */
X{    9, 20,     0,    3,  100,       0,  0x14 },  /* P-LFO Rate */
X{    9, 48,     0,    3,  100,       0,  0x15 },  /* P-LFO Depth */
X{    9, 75,     0,    3,  100,       0,  0x16 },  /* P-LFO Mod sens */
X/* Time Variable Filter */
X{   11, 48,     0,    3,  100,       0,  0x17 },  /* Cutoff Freq */
X{   11, 76,     0,    2,   30,       0,  0x18 },        /* Resonance */
X{   12, 21,     0,    2,   16,      -3,  0x19 },  /* Keyfollow */
```

```
X{   12, 48,     0,    3,  127,     -64,  0x1a },   /* bias point */
X{   12, 76,     0,    2,   14,      -7,  0x1b },   /* bias level */
X{   13, 20,     0,    3,  100,       0,  0x1c },   /* envelope depth */
X{   13, 48,     0,    3,  100,       0,  0x1d },   /* Velo Sens */
X{   13, 77,     0,    1,    4,       0,  0x1e },   /* Depth keyf */
X{   14, 22,     0,    1,    4,       0,  0x1f },   /* Time keyf */
X{   14, 48,     0,    3,  100,       0,  0x20 },   /* Time 1 */
X{   14, 75,     0,    3,  100,       0,  0x21 },   /* Time 2 */
X{   15, 20,     0,    3,  100,       0,  0x22 },   /* Time 3 */
X{   15, 48,     0,    3,  100,       0,  0x23 },   /* Time 4 */
X{   15, 75,     0,    3,  100,       0,  0x24 },   /* Time 5 */
X{   16, 20,     0,    3,  100,       0,  0x25 },   /* Level 1 */
X{   16, 48,     0,    3,  100,       0,  0x26 },   /* Level 2 */
X{   16, 75,     0,    3,  100,       0,  0x27 },   /* Level 3 */
X{   17, 20,     0,    3,  100,       0,  0x28 },   /* Sustain */
X/* Time Variable Amplifier */
X{   19, 48,     0,    3,  100,       0,  0x29 },   /* Main level */
X{   19, 75,     0,    3,  100,       0,  0x2a },   /* Velo Sens */
X{   20, 20,     0,    3,  127,     -64,  0x2b },   /* Bias 1 point */
X{   20, 48,     0,    3,   12,     -12,  0x2c },   /* Bias 1 level */
X{   20, 75,     0,    3,  127,     -64,  0x2d },   /* Bias 2 point */
X{   21, 20,     0,    3,   12,     -12,  0x2e },   /* Bias 2 level */
X{   21, 50,     0,    1,    4,       0,  0x2f },   /* Time keyfollow */
X{   21, 77,     0,    1,    4,       0,  0x30 },   /* Time velo keyf */
X{   22, 20,     0,    3,  100,       0,  0x31 },   /* Time 1 */
X{   22, 48,     0,    3,  100,       0,  0x32 },   /* Time 2 */
X{   22, 75,     0,    3,  100,       0,  0x33 },   /* Time 3 */
X{   23, 20,     0,    3,  100,       0,  0x34 },   /* Time 4 */
X{   23, 48,     0,    3,  100,       0,  0x35 },   /* Time 5 */
X{   23, 75,     0,    3,  100,       0,  0x36 },   /* Level 1 */
X{   24, 20,     0,    3,  100,       0,  0x37 },   /* Level 2 */
X{   24, 48,     0,    3,  100,       0,  0x38 },   /* Level 3 */
X{   24, 75,     0,    3,  100,       0,  0x39 }        /* Sustain */
X};
X
Xscreen_t timbrescreen = {
X    timbredata,
X    sizeof(timbredata)/sizeof(data_t),
X    timbrefield,
X    sizeof(timbrefield)/sizeof(field_t),
X    mtgetdata,
X    mtputdata,
X    mtkeys,
X    mtdone
X};
X
Xdata_t tdtmp[TDATAS];
Xfield_t tftmp[TFIELDS];
Xscreen_t tstmp;
X
Xdata_t commondata[] = {
X    {  1, 30, "MT32 Timbre Common Area" },
X    {  4, 20, "Tone Name   [" },
```

```
X    {  4, 43, "]" },
X    {  6, 20, "Structure Partials 1 and 2 (1-13)       [  ]" },
X    {  8, 20, "Structure Partials 3 and 4 (1-13)       [  ]" },
X    { 10, 20, "Partial Mute (T4, T3, T2, T1)           [    ]" },
X    { 12, 20, "Envelope Mode (0-normal, 1-no sustain)  [ ]" },
X    { 14, 20, "Partial to edit (0-none, 1-4)           [ ]" }
X};
X
Xfield_t commonfield[] = {
X/*   y   x   type   len   max   offset   disp */
X{    4, 33,     1,   10,    0,       0,     0 },  /* tone name */
X{    6, 61,     0,    2,   12,       1,    10 },  /* structure 1&2 */
X{    8, 61,     0,    2,   12,       1,    11 },  /* structure 3&4 */
X{   10, 61,     2,    4,   16,       0,    12 },  /* Mute */
X{   12, 61,     0,    1,    1,       0,    13 },  /* envelope mode */
X{   14, 61,     0,    1,    4,       0,    250 }  /* partial to edit */
X};
X
Xscreen_t commonscreen = {
X    commondata,
X    sizeof(commondata)/sizeof(data_t),
X    commonfield,
X    sizeof(commonfield)/sizeof(field_t),
X    mtgetdata,
X    mtputdata,
X    mtkeys,
X    mtdone
X};
X
Xstimbre()
X{
X    int s;
X    register field_t *fp;
X
X    mtwork[250] = 0;       /* partial to edit */
X    for (;;) {
X        if (screen(&commonscreen) < 0)
X            return;
X        switch (s = mtwork[250]) {
X        case 0:            /* no partial edits */
X            return;
X        case 1:
X        case 2:
X        case 3:
X        case 4:
X            s--;
X            s *= 58;      /* timbre data area */
X            s += 0x0e;         /* common data size */
X            memcpy(timbredata, tdtmp, TDATASIZ);
X            memcpy(timbrefield, tftmp, TFIELDSIZ);
X            memcpy(&timbrescreen, &tstmp, TSCREENSIZ);
X            tstmp.s_dp = tdtmp;
X            tstmp.s_fp = tftmp;
```

```
X             for (fp = tftmp; fp < &tftmp[TFIELDS]; fp++)
X                  fp->f_disp += s;
X             screen(&tstmp);
X         }
X    }
X}
SHAR_EOF
chmod 0444 timbre.c || echo "restore of timbre.c fails"
echo "x - extracting top.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > top.c &&
X#include <osbind.h>
X#include "mt.h"
X#include "mtscrn.h"
X#include "mttype.h"
X
Xuchar srctype;
Xuchar dsttype;
Xuchar srcentry;
Xuchar dstentry;
Xuchar srcfile[9];
Xuchar dstfile[9];
Xuchar edittype;
Xuchar midichan = 1;
X
Xdata_t topdata[] = {
X    { 1, 26, "MT32 Patch Editor" },
X    { 5, 12, "Src Type   [ ]  Entry #  [   ]  Src File  [" },
X    { 5, 63, "]" },
X    { 7, 12, "Dst Type   [ ]  Entry #  [   ]  Dst File  [" },
X    { 7, 63, "]" },
X    { 9, 16, "Type  1 - File" },
X    { 9, 40, "Midi Test Channel (1-16) [  ]" },
X    { 10, 22, "2 - MT32 Memory" },
X    { 12, 12, "Edit Type  [ ]    1 - Patch      (*.ptm)" },
X    { 13, 30, "2 - Patch Tmp  (*.ptt)" },
X    { 14, 30, "3 - Timbre     (*.tmb)" },
X    { 15, 30, "4 - Timbre Tmp (*.tmt)" },
X    { 16, 30, "5 - System     (*.sys)" },
X    { 17, 30, "6 - Setup Tmp  (*.set)" },
X    { 18, 30, "7 - Exit" }
X};
X
Xfield_t topfield[] = {
X    { 5, 24, 0, 1, 1, 1, 0 },
X    { 5, 38, 0, 3, 127, 1, 1 },
X    { 5, 55, 1, 8, 0, 0, 2 },
X    { 7, 24, 0, 1, 1, 1, 3 },
X    { 7, 38, 0, 3, 127, 1, 4 },
X    { 7, 55, 1, 8, 0, 0, 5 },
X    { 9, 66, 0, 2, 16, 1, 7 },
X    { 12, 24, 0, 1, 6, 1, 6 }
X};
X
```

```
Xuchar *topget();
Xint topput();
Xint mtkeys();
X
Xscreen_t topscrn = {
X    topdata,
X    sizeof(topdata)/sizeof(data_t),
X    topfield,
X    sizeof(topfield)/sizeof(field_t),
X    topget,
X    topput,
X    mtkeys,
X    topput
X};
X
Xtop()
X{
X    int ret;
X
X    edittype = 6;
X    for (;;) {
X         if (screen(&topscrn) < 0)
X              break;
X         if (edittype == 6)
X              break;
X         ksetcrsr(0, 20);
X         if (srctype == 0)  /* file */
X              ret = mtsrcf(edittype, srcfile);
X         else
X              ret = mtsrc(edittype, srcentry);
X         if (ret == -1) {
X              sleep(1);
X              continue;
X         }
X         if (dsttype == 0)  /* file */
X              ret = mtdstf(edittype, dstentry, dstfile);
X         else
X              ret = mtdst(edittype, dstentry);
X         if (ret == -1) {
X              sleep(1);
X              continue;
X         }
X         switch (edittype) {
X         case T_PATCH:          /* patch */
X              spatch();
X              break;
X         case T_PATCHTMP:       /* patch tmp */
X              spatchtmp();
X              break;
X         case T_TIMBRE:              /* timbre */
X         case T_TIMBRETMP:      /* timbre tmp */
X              stimbre();
X              break;
```

```
X         case T_SYSTEM:              /* system */
X              ssystem();
X              break;
X         case T_SETUPTMP:       /* setup tmp */
X              ssetuptmp();
X              break;
X         }
X    }
X    ksetcrsr(0, 23);
X    exit(0);
X}
X
Xuchar *
Xtopget(disp)
X{
X    switch (disp) {
X    case 0:
X         return(&srctype);
X         break;
X    case 1:
X         return(&srcentry);
X         break;
X    case 2:
X         return(srcfile);
X         break;
X    case 3:
X         return(&dsttype);
X         break;
X    case 4:
X         return(&dstentry);
X         break;
X    case 5:
X         return(dstfile);
X         break;
X    case 6:
X         return(&edittype);
X         break;
X    case 7:
X         return(&midichan);
X         break;
X    default:
X         printf("error\n");
X         exit(0);
X    }
X}
X
Xtopput()
X{
X}
X
SHAR_EOF
chmod 0444 top.c || echo "restore of top.c fails"
echo "x - extracting makefile (Text)"
```

```
sed 's/^X//' << 'SHAR_EOF' > makefile &&
XSRC =   makefile \
X    mt.h \
X    mtscrn.h \
X    mt.c \
X    mtscrn.c \
X    top.c \
X    src.c \
X    patch.c \
X    patchtmp.c \
X    timbre.c \
X    system.c \
X    setuptmp.c \
X    mttype.h \
X    keys.c
X
XOBJS =  mt.o \
X    mtscrn.o \
X    top.o \
X    src.o \
X    patch.o \
X    patchtmp.o \
X    timbre.o \
X    system.o \
X    setuptmp.o \
X    keys.o
X
Xmt32.tos: $(OBJS)
X    cc -o mt32.tos $(OBJS) -lar
X
Xcp:
X    cp $(SRC) a:\src\mt32
SHAR_EOF
chmod 0444 makefile || echo "restore of makefile fails"
echo "x - extracting mt32.doc (Text)"
sed 's/^X//' << 'SHAR_EOF' > mt32.doc &&
XDoc for MT32 patch editor:
X
XThe program is full screen, the editing keys are:
X
Xnumeric fields:
X    F1  -16
X    F2  -8
X    F3  -4
X    F4  -2
X    F5  -1
X    F6  +1
X    F7  +2
X    F8  +4
X    F9  +8
X    F10 +10
X
XAlphanumeric fields:
```

```
X    There is no backspace (sorry about the bug)
X    a-z, A-Z, and 0-9 are permitted.
X
XMoving from field to field is accomplished with the four
Xarrow keys.
X
XTo exit the current screen hit escape.  To excecute the current
Xscreen and go to a sub-screen hit return.
X
XWhen the output is directed to the mt32 the
Xpatch editor will edit in live mode:  every change
Xwill be reflected immediately.  Messages from a keyboard
Xare copied from the ST's MIDI in and sent the MIDI out
Xwhich allows a keyboard to be used to play the synth
Xwhile making changes to the patch.  Be warned that the mt32
Xwill reject patch changes while sounding a tone.  So all
Xnotes on the synth keyboard should be off when making changes
Xon the ST's keyboard.
X
XWhen the output is directed to the mt32 the second and third
Xrow of keys (q-] and a-') on the ST's keyboard form all of the
Xnotes in the C arpeggio for all 8 octaves.  The space bar sends
Xan all notes off msg.  While more privitive than having a full keyboard,
Xit allows for live editing without a keyboard.
XThe keys 1 through 8 allow one to change velocity
Xsetting on the notes generated from thw ST's keyboard.
X
XCabling is difficult, there is no setup that lets you source
Xand sink to the mt32 and have a synth keyboard hooked up
Xall at the same time.  You get one or the other but not
Xboth.  To do live editing you need the MT-32's out connected
Xto the ST's in and the ST's out connected to the MT-32's in.
X
XFor synth keyboard usage, use the above cabling method first and
Xcopy the patches to disk files.  Then switch cables so that the
Xsynth kbd is connected to the ST's in.  The MT-32's out will
Xbe disconnected.  When done editing the new patch will be in the
Xmt-32's memory.  Another cable switch and you can copy the new patch
Xto a disk file.
X
XWhat all the patch fields are:
XAll of my names closely track Roland's names.
XThe mt32 and the D-50 have similar names for everything.
XI suggest a D-50 book if you can't find an mt32 book at
Xyour local music store.  The D-10/20/110 changed the names
Xfor everything, so while very similar, reading such doc
Xwill be confusing.
X
XAlso, I recommend reading the MIDI implementation in the back
Xfor sysex messages.  This is where I got my names and where you
Xcan find out which kind of memory does what.
X
XFile formats:
```

```
XMy file formats are exactly the sysex message it would take to
Xsend the data to the mt32.  Thus, if your sequencer allows importing
Xsuch files, you can store song setup information in your seqence.
XThe mt-32 needs about a half a second to digest each such message.
XI did not include a program that would dump sysex files to the
XMIDI port.  Such a program is trivial (for programmers) and I might
Xdo it for those interested.
X
X             - kurt
X
XOn compiling the source:
X    the memcpy.s file is not referenced in the makefile.
X    It is the only item needed from the archive libar.a
X    (-lar).
SHAR_EOF
chmod 0444 mt32.doc || echo "restore of mt32.doc fails"
exit 0


From 100051122920113572
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,e6e55a1889c25322,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-01-19 00:59:08 PST
Path: utzoo!attcan!uunet!lll-winken!ncis.llnl.gov!ncis!helios.ee.lbl.gov!pasteur!
ucbvax!agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i092: starchrt -- Diffs to Unix starchart program for ST  part01/02
Keywords: shar, diff, C
Message-ID: <6027@saturn.ucsc.edu>
Date: 19 Jan 89 08:59:08 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 998
Approved: koreth@ssyx.ucsc.edu

Submitted-by: cs.buffalo.edu!sigmast!dgy
Posting-number: Volume 1, Issue 92
Archive-name: starchrt/part01

This archive contains the patches needed to get starchart 2.0 running on the
ST, along with the driver for the ST's color monitor and the utility programs
provided with the companion posting in comp.binaries.atari.st.

#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
# This is part 1 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
#
#    Run the following text with /bin/sh to create:
#      Makefile.st
#      README
```

```
#      README.2
#      moonphse.dif
#      planet.dif
#      staranim.c
#      starargs.c
#      starchrt.dif
#      starlase.dif
#      starpost.dif
#      starst.c
#
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
sed 's/^X//' << 'SHAR_EOF' > Makefile.st &&
X#
X# Author: Petri Launiainen, Intrinsic Oy, Tampere, FINLAND (pl@intrin.FI)
X#
X# Based on the work of Alan W. Paeth
X#
X# Modified by Dave Yearke (dgy@sigmast) for the Atari ST,
X# using Mark Williams C, version 3.0.6.
X
XLOCAL=-DST_MWC
X#
X# put the Yale Star Catalog file path here: because the yale.star is
X# static (stars don't change so often :->), consider using some static
X# library path here - relative one is used for quick testing
X
XSTAR = "yale.str"
X
X# similarly assign the Messier file path:
X
XMESS = "messier.str"
X
X# and the constellation finder file:
X
XCONS = "con.loc"
X
X# put the planet position file path here: it should be preferably
X# relative, thus allowing everyone to change planet positions according
X# to their interests.  This file changes when the 'planet' program is
X# executed
X
XPLANET = "planet.str"
X
X# notice that starchart.o and planet are made dependent on Makefile, due
X# to the STARFILE, MESSFILE, CONSTFILE and PLANETFILE definitions here
X
XCFLAGS = -VCOMPAC -VPEEP -DSTARFILE='$(STAR)' -DPLANETFILE='$(PLANET)' \
X         -DCONSTFILE='$(CONS)' -DMESSFILE='$(MESS)' $(LOCAL)
X# MWC needs -f to include floating point printf() routines
XLFLAGS = -s -f
```

```
XLIBS= -lm
X
X# star chart calculation routine file
X
XCOBJ = starchrt.o
X
X# default binary names: if you change these, change also the corresponding
X# make tags
X
XBINS = epoch.ttp moonphse.tos planet.ttp starlase.ttp starpost.ttp starst.ttp
X
Xtest:
X    starst -c ori
X
Xall:    \
X    planet.ttp\
X    starpost.ttp\
X    starlase.ttp\
X    starst.ttp\
X    epoch.ttp\
X    moonphse.tos
X    @echo Everything has been compiled.
X
Xstarchrt.o: starchrt.c Makefile starchrt.h
X
Xmoonphse.tos: moonphse.c
X    cc $(CFLAGS) $(LFLAGS) moonphse.c $(LIBS) -o $@
X
Xepoch.ttp: epoch.c
X    cc $(CFLAGS) $(LFLAGS) epoch.c $(LIBS) -o $@
X
X# DO NOT use -DCOMPAC on planet or the linker will go nuts!
Xtest.ttp: test.c Makefile
X    cc $(LFLAGS) $(LOCAL) -DPLANETFILE='$(PLANET)' -VPEEP \
X       test.c $(LIBS) -o $@
X
Xplanet.ttp: planet.c Makefile
X    cc $(LFLAGS) $(LOCAL) -DPLANETFILE='$(PLANET)' -VPEEP \
X       planet.c $(LIBS) -o $@
X
Xstarlase.ttp: $(COBJ) starlase.o starimag.o
X    cc $(LFLAGS) $(COBJ) starlase.o starimag.o $(LIBS) -o $@
X
Xstarpost.ttp: $(COBJ) starpost.o
X    cc $(LFLAGS) $(COBJ) starpost.o $(LIBS) -o $@
X
X# Recompile starchrt.o to elide the die() function.
Xstarst.ttp: starst.o
X    cc $(CFLAGS) -DINTERACTIVE -c starchrt.c
X    cc $(LFLAGS) $(COBJ) starst.o $(LIBS) -laes -lvdi -o $@
X    rm starchrt.o
X
Xstarargs.tos: starargs.c
```

```
X    cc -s -VSMALL -VPEEP starargs.c -o $@
X
Xstaranim.ttp: staranim.c
X    cc -s -VSMALL -VPEEP staranim.c -o $@
Xclean:
X    rm -f *.o mlog core
SHAR_EOF
chmod 0600 Makefile.st || echo "restore of Makefile.st fails"
sed 's/^X//' << 'SHAR_EOF' > README &&
XA hack is used to determine if "moonphse," "planet," "starst," or "staranim"
Xis running from the desktop or the shell.  If the environment variable "PATH"
Xis not set or has a null value, the program assumes it was run from the
Xdesktop and will wait for a keypress before exiting.  If "PATH" is set, it
Xassumes it was run from a shell (since "PATH" is meaningless to the desktop)
Xand will usually exit immediately, the exception being "STARST.TTP" as
Xdescribed in the readme file for the executables.
X
XTwo defines were added to starchart:  ST_MWC defines the Atari ST with the
XMark Williams C compiler, and INTERACTIVE means that the program must reset
Xthe terminal before exiting, so the function die() is located in the driver
Xsource file instead of in starchrt.c for better portability.
X
XReally, very few changes were necessary.  A couple of frustrating bugs turned
Xout to be caused by Mark Williams C's definition of an "int" as 16 bits, which
Xcaused very large numbers to become smaller, negative ones.  Casting them to
X"long" solved the problem, but, as usual, fixing the problem was much easier
Xthan finding it.
X
XNames of programs were changed as follows because of the ST's MS-DOS-like file
Xstructure:
X     moonphase.c --> moonphse.c
X     starchart.c --> starchrt.c
X     starchart.h --> starchrt.h
X     starimages.c --> starimag.c
X     starlaser.c --> starlase.c
X
XBe warned that compiling "planet" requires almost a full megabyte of memory.
XWith my 1040 I had to remove ALL desk accessories and AUTO folder programs,
Xand even then had to redirect the output of "make -n planet.ttp" into a file
Xand invoke it as a shell script, due to the overhead of running "make."
X
XSee the file "readme.2" for the readme files that were included with the
Xoriginal distributions.
X
XObligatory Caveat:  I make no guarantees as to the robustness or accuracy of
Xthis port of starchart.  While I am sure there are no devestating bugs in it,
Xshould one appear I am not responsible for any damage or misinformation it
Xmay cause.
X
XAll programs were compiled with Mark Williams C, version 3.0.6; portions of
Xthese programs copyright 1984, Mark Williams Company.
X
Xquestions or comments on the ST version only should go to:
```

```
Xdgy@sigmast.UUCP (Dave Yearke, Sigma Systems Technology, Inc.)
SHAR_EOF
chmod 0600 README || echo "restore of README fails"
sed 's/^X//' << 'SHAR_EOF' > README.2 &&
XREADME1 ========================================================================
X
XSTARCHART (rev 2)
X-------------------------------------------------------------------------------
XNote: This version is upward compatable with the original "starchart" software
Xfamily, both in user interface and in continued support of the original
Xreduced Yale catalog. Thus, it may supplant old copies of the software without
Xfear that either shell scripts or file formats will become invalid.
X
XHowever, both the new user interface and the extended catalog are
Xsuperior, as are extensions for additional labelling (Greek characters,
Xdouble star indicators, Messier objects), plus a mnemonic "finder" facility.
X-------------------------------------------------------------------------------
X
XINSTALLATION INSTRUCTIONS:
X
X[1]
XSave the new posting of the reduced Yale Star Catalog under the name
X"yale.star" (or use a copy of the old version). If the save directory is other
Xthan the one in which the program lives, then check out the Makefile, which
Xallows path names to be specified for compilation.
X
XSites without much disk space may truncate the file yale.star at any point,
Xas the data is sorted in order of increasing magnitude (dimming).
X
XAdditonally, the optional files "messier.star" and "con.locs" should be
Xplaced in the chosen directory. Entries in these files are unsorted. The
Xoptional file "planet.star" is created on an as-needed basis by the softwaare
X"planet.c", and might want a home in some less permanent (writable) location.
X
X[2]
XDo a "make all" to compile all software. They are described in the man pages
X"starchart.1" and "planet.1" . All compile using starchart.c as the mainline,
Xwith device driver specifics in modules of appropriate name.
X
X[3]
XYou may now try "man test" to make an Orion pic file similar to the one
Xposted to the net. Or you may test by outputting to the tty display, using the
Xversion "stardsp". This is described in the EXAMPLE section of the man page.
X
XVERSION NOTES:
X
XTty display users (*sigh*) may which to alter "stardsp" to output
Xto the display using an appropriate number of of rows and columns
Xon the screen. (Presently 31x79 for a 32x80 display. 23x79 is a
Xbit sparse).  Find the ROWS and COLS #defines in stardsp.c
Xand update. Or be really ambitious and recode them as run time
Xvariables.
X
```

XTektronix emulator users may wish to alter scaling parameters based on the
Xactual physical dimensions of the display raster, to get the cleanest possible
Xstar glyphs on output. This is described at length in the beginning of the
Xdriver "startek.c".
X
XPic file users: the #define PICFRAG can be adjusted downward
X(presently 8) should your pic input overflow pic's buffers. This
Xhappens when a long move,draw,draw,...,draw sequence happens.
XSmaller PICFRAGs make larger files, by forming fragmented
Xmove,draw commands.
X
XPostScript users: you may wish to play with the boiler-plate macros which
Xdefine star shapes (e.g. 's1' does first magnitude stars). This is mentioned
Xat the end of starchart.c
X
XLaserjet users: the output of starlaser must be directed to
Xlaserjet via raw connection, because it is largely sent in binary
X8-bit format. The graphics memory might overflow if you use the
Xstandard (oldest) 59k buffer laserjet (*sigh*). Anyway, I haven't
Xever lost any relevant data on the sheet: the overflow has occurred on
Xthe last few blank pixel rows of the graphics area. If you use
Xlaserjet plus, you shouldn't have any problems.
X
XOther devices: don't expect the authors to write a new driver for you :-)
XLook through the sources - it's not so difficult after all... The module
X"starimages.c" provides support for bitmap devices; the PostScript driver
Xis a good example of generating output in some graphics language.
X
XMore ambitious programmers may wish to extend the usefulness of these programs.
XBe our guest, but carefully consider the proposed list of suggestions which
Xintentionally accompanies the main source code body "starchart.c". It is hoped
Xthat future changes and major revisions and their subsequent integration will
Xbe kept relatively simple by coordinating efforts and by not duplicating work.
X
Xawpaeth@watcgl (original author and revisions editor)
Xinp@amber.berkeley.EDU (revisionist for user interface)
Xpl@sibelius.intrin.FI (support for mini-Yale, Messier and finder catalogs)
X
XREADME2 ==
X
X[from awpaeth:]
XThis is the SOFTWARE update kit for StarChart version #2
Xwhich was released in Nov/Dec 1987. The kit performs an upgrade to ver #2.1
XNo upgrades to yale.star or other .star files occurs.
X
XThis kit updates StarChart version #2 software as it appeared in that release -
Xno other patches should already be in place. Virgin copies of version #2 are
Xavailable from the moderator of comp.sources.unix.
X
X
XOverview of Changes
X
XMore portability across sysV and IRIS UNIX installations.

```
XA particularly nasty bug in planet.c was removed.
XTwo new device drivers were added (Unix Plot(5) and Iris).
XStarChart now performs full clipping of vectors.
XAdditional flags for creating stylized output and larger displays.
XPostScript now has nicer symbols for non-stellar objects.
XMore detailed patch and upgrade notes appear in the source code.
X
XFILES
X
XName              Description/Instructions
X----              ------------------------
XREADME             this file
Xpatch-*            patches for moonphase.c, planet.c, drivers, yaleformat
Xpatchall      runs the above patches on the originals
Xstarchart.1        new man page
Xstarchart.c        rewritten main driver (easier to reinstall then patch)
Xstarchart.h        accompanying new .h file for above (ditto)
Xstariris.c         new IRIS driver
Xstaruplot.c        new UNIX Plot(5) driver
X
XREADME3 ========================================================================
X
XOver the last few weeks I have generated a database of constellation
Xboundaries.  This database does NOT include the constellation figures, just
Xthe boundaries defining the area of the sky the constellation contains.
XThis database is in the 'old starchart' format, thus it will work for the
Xrecently posted starchart version 2.0 and the original starchart 1.0.
X
XThis data was hand compiled from the Sky_Atlas_2000.0.  Each constellation
Xhas it's entire boundary defined, thus each adjacent constellation will define
Xthe same vector (it will exist twice in the database).
X
XTo make this work for the starchart program, a modification is needed in the
X'chartobjects' procedure.  I have included the entire procedure in the
Xshar file following this message.  The only changes were at the top where
XI added some simple window clipping checks for the vectors and a couple
Xof new variables.  Any vector outside the defined window will be converted
Xto a 'move' command and placed at the window edge.  You can either replace
Xthe entire procedure or just make the necessary changes.
X
XIf you wish the boundaries to always be charted, you should modify the
Xstarchart.c module to read in and chart the boundary.star file just like
Xthe other databases.  If you only rarely chart them, then just use the
X'-f' option in starchart version 2.0
X
XHave fun.
XDave
X
XUUCP: {decvax,ucbvax,ihnp4,hplabs,sun,ames,prls}!decwrl!mips!vanthof
Xor    vanthof@mips.com     (really Dave Van't Hof)
XUSPS: MIPS Computer Systems, 930 Arques, Sunnyvale, CA 94086, (408) 991-0242
X"Always? No. Never question authority."  What I say is mine.
SHAR_EOF
```

```
chmod 0600 README.2 || echo "restore of README.2 fails"
sed 's/^X//' << 'SHAR_EOF' > moonphse.dif &&
X*** ../starchart/moonphase.c    Mon Sep 19 09:11:06 1988
X--- moonphse.c    Mon Sep 26 08:00:23 1988
X***************
X*** 12,17 ****
X--- 12,21 ----
X  ! modified by awpaeth@watcgl, December 1987 for sysV compatability
X  ******************************************************************************/
X
X+ #ifdef ST_MWC /* Mark Williams C for the Atari ST */
X+ #define SYSV /* MWC is almost SYSV compatible */
X+ #endif  /* ST_MWC */
X+
X  #include <stdio.h>
X  #ifndef SYSV
X  #include <sys/time.h>
X***************
X*** 19,25 ****
X--- 23,35 ----
X  #include <time.h>
X  #endif
X  #include <math.h>
X+ #ifdef ST_MWC
X+ #include <osbind.h> /* Needed for Cconws() and Crawcin() */
X+ #define cfree() free() /* MWC doesn't have cfree() */
X+ #endif /* ST_MWC */
X+ #ifndef PI
X  #define PI         3.141592654
X+ #endif
X  #define EPOCH   1983
X  #define EPSILONg 279.103035     /* solar ecliptic long at EPOCH */
X  #define RHOg     282.648015     /* solar ecliptic long of perigee at EPOCH */
X***************
X*** 34,40 ****
X--- 44,55 ----
X  double adj360();
X  double potm();
X
X+ #ifdef ST_MWC
X+ extern char *getenv(); /* Used to see if we're running from the desktop */
X+ time_t lo;
X+ #else /* !ST_MWC */
X  long *lo = (long *) calloc (1, sizeof(long)); /* used by time calls */
X+ #endif /* ST_MWC */
X  struct tm *pt; /* ptr to time structure */
X
X  double days;   /* days since EPOCH */
X***************
X*** 42,50 ****
X--- 57,70 ----
X  double phase2; /* percent of lunar surface illuminated one day later */
X  int i = EPOCH;
```

```
X
X+ #ifdef ST_MWC
X+ time (&lo);  /* get system time */
X+ pt = gmtime(&lo);  /* get ptr to gmt time struct */
X+ #else /* !ST_MWC */
X  time (lo);  /* get system time */
X  pt = gmtime(lo);  /* get ptr to gmt time struct */
X  cfree(lo);
X+ #endif /* ST_MWC */
X
X  /* calculate days since EPOCH */
X  days = (pt->tm_yday +1) + ((pt->tm_hour + (pt->tm_min / 60.0)
X***************
X*** 82,87 ****
X--- 102,115 ----
X        printf("Waning ");
X     printf("Crescent (%1.0f%% of Full)\n", phase);
X     }
X+ #ifdef ST_MWC /* We want to hold the screen if invoked from the GEM desktop; */
X+            /* the desktop doesn't usually set any environment variables, */
X+            /* and if getenv() does find anything, it's probably empty */
X+ if (((getenv("PATH")) == 0) || (strlen(getenv("PATH")) == 0)) {
X+      Cconws("press any key to continue: "); /* Hold screen if desktop */
X+      i = Crawcin(); /* Read raw character input */
X+ }
X+ #endif /* ST_MWC */
X  }
X
X  double potm(days)
SHAR_EOF
chmod 0600 moonphse.dif || echo "restore of moonphse.dif fails"
sed 's/^X//' << 'SHAR_EOF' > planet.dif &&
X*** ../starchart/planet.c  Mon Sep 19 09:11:43 1988
X--- planet.c Mon Sep 26 08:00:31 1988
X***************
X*** 45,57 ****
X  #include <stdio.h>
X  #include <math.h>
X
X  #ifndef SYSV
X  #include <sys/time.h>    /* for getting current GMT (generic Unix) */
X  #else
X  #include <time.h>   /* for getting current GMT (sysV version) */
X  #endif
X
X! #define CURYEAR 1987 /* default year -- needs to be maintained */
X
X  #ifndef PLANETFILE
X  #define PLANETFILE "./planet.star"
X--- 45,62 ----
X  #include <stdio.h>
X  #include <math.h>
X
```

```
X+ #ifdef ST_MWC /* Mark Williams C for the Atari ST */
X+ #define SYSV /* MWC is almost SYSV compatible */
X+ #include <osbind.h> /* Needed for Cconws() and Crawcin() */
X+ #endif /* ST_MWC */
X+
X  #ifndef SYSV
X  #include <sys/time.h>    /* for getting current GMT (generic Unix) */
X  #else
X  #include <time.h>    /* for getting current GMT (sysV version) */
X  #endif
X
X! #define CURYEAR 1989 /* default year -- needs to be maintained */
X
X  #ifndef PLANETFILE
X  #define PLANETFILE "./planet.star"
X***************
X*** 68,73 ****
X--- 73,84 ----
X  #define MAGURA 590
X  #define MAGNEP 800
X
X+ #ifdef ST_MWC
X+ #define daylight tt.tm_isdst
X+ extern char *getenv(); /* used to see if we're running from the desktop */
X+ struct tm_t *tt;
X+ #endif /* ST_MWC */
X+
X  double pie, rad;
X  double htod(), atof(), kepler(), truean();
X  double longi(), lati(), poly(), aint(), range();
X***************
X*** 119,125 ****
X--- 130,138 ----
X      gettimeofday(&tv, &tz);
X      return(GMT1970 + tv.tv_sec/SECSPERDAY);
X  #else
X+ #ifndef ST_MWC
X      long time();
X+ #endif /* ST_MWC */
X      return(GMT1970 + (double)time((long *)0)/SECSPERDAY);
X  #endif
X      }
X***************
X*** 201,207 ****
X      aa1 = (int) (year/100);
X      bb1 = 2 - aa1 + (int)(aa1/4);
X      jd = aint(365.25*year) + aint(30.6001*(month + 1));
X!     jd = jd  + day + 1720994.5;
X      if((year + month/100) > 1582.10) jd = jd + bb1;
X      return(jd);
X      }
X--- 214,220 ----
X      aa1 = (int) (year/100);
```

```
X      bb1 = 2 - aa1 + (int)(aa1/4);
X      jd = aint(365.25*year) + aint(30.6001*(month + 1));
X!     jd += day + 1720994.5;
X      if((year + month/100) > 1582.10) jd = jd + bb1;
X      return(jd);
X      }
X***************
X*** 218,224 ****
X--- 231,243 ----
X    double N,D,epli,thapp,omeg;
X    double nu2,P,Q,S,V,W,ze,l1pert,epert,w1pert,apert;
X    double psi,H,G,eta,th;
X+ #ifdef ST_MWC
X+   time_t now;
X
X+   now = time((long) 0);
X+   tt = localtime(&now);
X+ #endif /* ST_MWC */
X+
X  #define WRITEMODE "w"
X  #define OPENFAIL 0
X    progname = argv[0];
X***************
X*** 321,327 ****
X         RA = atan2(N,D)/rad;
X         DEC = asin(sin(epli*rad)*sin(thapp*rad))/rad;
X         speak(RA,DEC,Sr, MAGSOL, "PS", "Sol");
X! /* tansformation of coordinates on Mercury and output */
X         trans(r,b,ll,Stheta,Sr,epli, MAGMER, "PM", "Mercury");
X
X  /* Now start on Venus */
X--- 340,346 ----
X         RA = atan2(N,D)/rad;
X         DEC = asin(sin(epli*rad)*sin(thapp*rad))/rad;
X         speak(RA,DEC,Sr, MAGSOL, "PS", "Sol");
X! /* transformation of coordinates on Mercury and output */
X         trans(r,b,ll,Stheta,Sr,epli, MAGMER, "PM", "Mercury");
X
X  /* Now start on Venus */
X***************
X*** 373,380 ****
X    radpert = 0.000022501 * cos((2*M-2*M2-58.208)*rad)
X          +0.000019045 * cos((3*M-3*M2+92.577)*rad)
X          +0.000006887 * cos((M5-M2-118.090)*rad)
X!         +0.000005172 * cos((M-M2-29.110)*rad)
X!         +0.000003620 * cos((5*M-4*M2-104.208)*rad)
X          +0.000003283 * cos((4*M-4*M2+63.513)*rad)
X          +0.000003074 * cos((2*M5-2*M2-55.167)*rad);
X    r = r + radpert;
X--- 392,399 ----
X    radpert = 0.000022501 * cos((2*M-2*M2-58.208)*rad)
X          +0.000019045 * cos((3*M-3*M2+92.577)*rad)
X          +0.000006887 * cos((M5-M2-118.090)*rad)
```

```
X!        +0.000005172 * cos((M-M2-29.110)*rad);
X!   radpert+= 0.000003620 * cos((5*M-4*M2-104.208)*rad)
X         +0.000003283 * cos((4*M-4*M2+63.513)*rad)
X         +0.000003074 * cos((2*M5-2*M2-55.167)*rad);
X    r = r + radpert;
X***************
X*** 425,432 ****
X    lonpert = 0.00705*cos((M5-M4-48.958)*rad)
X         +0.00607*cos((2*M5-M4-188.350)*rad)
X         +0.00445*cos((2*M5-2*M4-191.897)*rad)
X!        +0.00388*cos((M-2*M4+20.495)*rad)
X!        +0.00238*cos((M-M4+35.097)*rad)
X         +0.00204*cos((2*M-3*M4+158.638)*rad)
X         +0.00177*cos((3*M4-M2-57.602)*rad)
X         +0.00136*cos((2*M-4*M4+154.093)*rad)
X--- 444,451 ----
X    lonpert = 0.00705*cos((M5-M4-48.958)*rad)
X         +0.00607*cos((2*M5-M4-188.350)*rad)
X         +0.00445*cos((2*M5-2*M4-191.897)*rad)
X!        +0.00388*cos((M-2*M4+20.495)*rad);
X!   lonpert+= 0.00238*cos((M-M4+35.097)*rad)
X         +0.00204*cos((2*M-3*M4+158.638)*rad)
X         +0.00177*cos((3*M4-M2-57.602)*rad)
X         +0.00136*cos((2*M-4*M4+154.093)*rad)
X***************
X*** 445,452 ****
X    radpert+= 0.000007914*cos((3*M5-2*M4-139.737)*rad)
X         +0.000007004*cos((2*M5-3*M4-102.888)*rad)
X         +0.000006620*cos((M-2*M4+113.202)*rad)
X!        +0.000004930*cos((3*M5-3*M4-76.243)*rad)
X!        +0.000004693*cos((3*M-5*M4+190.603)*rad)
X         +0.000004571*cos((2*M-4*M4+244.702)*rad)
X         +0.000004409*cos((3*M5-M4-115.828)*rad);
X    r = r + radpert;
X--- 464,471 ----
X    radpert+= 0.000007914*cos((3*M5-2*M4-139.737)*rad)
X         +0.000007004*cos((2*M5-3*M4-102.888)*rad)
X         +0.000006620*cos((M-2*M4+113.202)*rad)
X!        +0.000004930*cos((3*M5-3*M4-76.243)*rad);
X!   radpert+= 0.000004693*cos((3*M-5*M4+190.603)*rad)
X         +0.000004571*cos((2*M-4*M4+244.702)*rad)
X         +0.000004409*cos((3*M5-M4-115.828)*rad);
X    r = r + radpert;
X***************
X*** 507,513 ****
X    -(0.003083 + 0.000275*nu2 - 0.000489*nu2*nu2)*sin(2*V)
X    +0.002472*sin(W)
X    +0.013619*sin(ze)
X!   +0.018472*sin(2*ze)
X    +0.006717*sin(3*ze)
X    +0.002775*sin(4*ze)
X    +(0.007275 - 0.001253*nu2)*sin(ze)*sin(Q)
X--- 526,533 ----
```

```
X    -(0.003083 + 0.000275*nu2 - 0.000489*nu2*nu2)*sin(2*V)
X    +0.002472*sin(W)
X    +0.013619*sin(ze)
X!   +0.018472*sin(2*ze);
X! l1pert = l1pert
X    +0.006717*sin(3*ze)
X    +0.002775*sin(4*ze)
X    +(0.007275 - 0.001253*nu2)*sin(ze)*sin(Q)
X***************
X*** 519,525 ****
X    -(0.035681 + 0.001208*nu2)*sin(ze)*cos(Q)
X    -0.004261*sin(2*ze)*cos(Q)
X    +0.002178*cos(Q)
X!   +(-0.006333 + 0.001161*nu2)*cos(ze)*cos(Q)
X    -0.006675*cos(2*ze)*cos(Q)
X    -0.002664*cos(3*ze)*cos(Q)
X    -0.002572*sin(ze)*sin(2*Q)
X--- 539,546 ----
X    -(0.035681 + 0.001208*nu2)*sin(ze)*cos(Q)
X    -0.004261*sin(2*ze)*cos(Q)
X    +0.002178*cos(Q)
X!   +(-0.006333 + 0.001161*nu2)*cos(ze)*cos(Q);
X! l1pert = l1pert
X    -0.006675*cos(2*ze)*cos(Q)
X    -0.002664*cos(3*ze)*cos(Q)
X    -0.002572*sin(ze)*sin(2*Q)
X***************
X*** 532,538 ****
X    -.0006764*sin(ze)*sin(Q)
X    -.0001110*sin(2*ze)*sin(Q)
X    -.0000224*sin(3*ze)*sin(Q)
X!   -.0000204*sin(Q)
X    +(.0001284 + .0000116*nu2)*cos(ze)*sin(Q)
X    +.0000188*cos(2*ze)*sin(Q)
X    +(.0001460 + .0000130*nu2)*sin(ze)*cos(Q)
X--- 553,560 ----
X    -.0006764*sin(ze)*sin(Q)
X    -.0001110*sin(2*ze)*sin(Q)
X    -.0000224*sin(3*ze)*sin(Q)
X!   -.0000204*sin(Q);
X! epert = epert
X    +(.0001284 + .0000116*nu2)*cos(ze)*sin(Q)
X    +.0000188*cos(2*ze)*sin(Q)
X    +(.0001460 + .0000130*nu2)*sin(ze)*cos(Q)
X***************
X*** 544,550 ****
X    +.0000508*cos(3*ze)*cos(Q)
X    +.0000230*cos(4*ze)*cos(Q)
X    +.0000108*cos(5*ze)*cos(Q)
X!   -(.0000956 + .0000073*nu2)*sin(ze)*sin(2*Q)
X    +.0000448*sin(2*ze)*sin(2*Q)
X    +.0000137*sin(3*ze)*sin(2*Q)
X    +(-.0000997 + .0000108*nu2)*cos(ze)*sin(2*Q)
```

```
X--- 566,573 ----
X    +.0000508*cos(3*ze)*cos(Q)
X    +.0000230*cos(4*ze)*cos(Q)
X    +.0000108*cos(5*ze)*cos(Q)
X!   -(.0000956 + .0000073*nu2)*sin(ze)*sin(2*Q);
X! epert = epert
X    +.0000448*sin(2*ze)*sin(2*Q)
X    +.0000137*sin(3*ze)*sin(2*Q)
X    +(-.0000997 + .0000108*nu2)*cos(ze)*sin(2*Q)
X***************
X*** 554,560 ****
X    +(-.0000956 +.0000099*nu2)*sin(ze)*cos(2*Q)
X    +.0000490*sin(2*ze)*cos(2*Q)
X    +.0000158*sin(3*ze)*cos(2*Q)
X!   +.0000179*cos(2*Q)
X    +(.0001024 + .0000075*nu2)*cos(ze)*cos(2*Q)
X    -.0000437*cos(2*ze)*cos(2*Q)
X    -.0000132*cos(3*ze)*cos(2*Q);
X--- 577,584 ----
X    +(-.0000956 +.0000099*nu2)*sin(ze)*cos(2*Q)
X    +.0000490*sin(2*ze)*cos(2*Q)
X    +.0000158*sin(3*ze)*cos(2*Q)
X!   +.0000179*cos(2*Q);
X! epert = epert
X    +(.0001024 + .0000075*nu2)*cos(ze)*cos(2*Q)
X    -.0000437*cos(2*ze)*cos(2*Q)
X    -.0000132*cos(3*ze)*cos(2*Q);
X***************
X*** 562,568 ****
X  w1pert = (0.007192 - 0.003147*nu2)*sin(V)
X    +(-0.020428 - 0.000675*nu2 + 0.000197*nu2*nu2)*cos(V)
X    +(0.007269 + 0.000672*nu2)*sin(ze)*sin(Q)
X!   -0.004344*sin(Q)
X    +0.034036*cos(ze)*sin(Q)
X    +0.005614*cos(2*ze)*sin(Q)
X    +0.002964*cos(3*ze)*sin(Q)
X--- 586,593 ----
X  w1pert = (0.007192 - 0.003147*nu2)*sin(V)
X    +(-0.020428 - 0.000675*nu2 + 0.000197*nu2*nu2)*cos(V)
X    +(0.007269 + 0.000672*nu2)*sin(ze)*sin(Q)
X!   -0.004344*sin(Q);
X! w1pert = w1pert
X    +0.034036*cos(ze)*sin(Q)
X    +0.005614*cos(2*ze)*sin(Q)
X    +0.002964*cos(3*ze)*sin(Q)
X***************
X*** 573,579 ****
X    -0.006603*cos(ze)*cos(Q)
X    -0.005356*sin(ze)*sin(2*Q)
X    +0.002722*sin(2*ze)*sin(2*Q)
X!   +0.004483*cos(ze)*sin(2*Q)
X    -0.002642*cos(2*ze)*sin(2*Q)
X    +0.004403*sin(ze)*cos(2*Q)
```

```
X    -0.002536*sin(2*ze)*cos(2*Q)
X--- 598,605 ----
X    -0.006603*cos(ze)*cos(Q)
X    -0.005356*sin(ze)*sin(2*Q)
X    +0.002722*sin(2*ze)*sin(2*Q)
X!   +0.004483*cos(ze)*sin(2*Q);
X! w1pert = w1pert
X    -0.002642*cos(2*ze)*sin(2*Q)
X    +0.004403*sin(ze)*cos(2*Q)
X    -0.002536*sin(2*ze)*cos(2*Q)
X***************
X*** 591,597 ****
X    +.000693*cos(2*ze)
X    +.000312*cos(3*ze)
X    +.000147*cos(4*ze)
X!   +.000299*sin(ze)*sin(Q)
X    +.000181*cos(2*ze)*sin(Q)
X    +.000204*sin(2*ze)*cos(Q)
X    +.000111*sin(3*ze)*cos(Q)
X--- 617,624 ----
X    +.000693*cos(2*ze)
X    +.000312*cos(3*ze)
X    +.000147*cos(4*ze)
X!   +.000299*sin(ze)*sin(Q);
X! apert = apert
X    +.000181*cos(2*ze)*sin(Q)
X    +.000204*sin(2*ze)*cos(Q)
X    +.000111*sin(3*ze)*cos(Q)
X***************
X*** 645,651 ****
X  l1pert = (-0.814181 + 0.018150*nu2 + 0.016714*nu2*nu2)*sin(V)
X     +(-0.010497 + 0.160906*nu2 - 0.004100*nu2*nu2)*cos(V)
X     +0.007581*sin(2*V)
X!    -0.007986*sin(W)
X     -0.148811*sin(ze)
X     -0.040786*sin(2*ze)
X     -0.015208*sin(3*ze)
X--- 672,679 ----
X  l1pert = (-0.814181 + 0.018150*nu2 + 0.016714*nu2*nu2)*sin(V)
X     +(-0.010497 + 0.160906*nu2 - 0.004100*nu2*nu2)*cos(V)
X     +0.007581*sin(2*V)
X!    -0.007986*sin(W);
X! l1pert = l1pert
X     -0.148811*sin(ze)
X     -0.040786*sin(2*ze)
X     -0.015208*sin(3*ze)
X***************
X*** 655,661 ****
X    +(0.008931 + 0.002728*nu2)*sin(ze)*sin(Q)
X    -0.016500*sin(2*ze)*sin(Q)
X    -0.005775*sin(3*ze)*sin(Q)
X!   +(0.081344 + 0.003206*nu2)*cos(ze)*sin(Q)
X    +0.015019*cos(2*ze)*sin(Q)
```

```
X    +(0.085581 + 0.002494*nu2)*sin(ze)*cos(Q)
X    +(0.025328 - 0.003117*nu2)*cos(ze)*cos(Q);
X--- 683,690 ----
X    +(0.008931 + 0.002728*nu2)*sin(ze)*sin(Q)
X    -0.016500*sin(2*ze)*sin(Q)
X    -0.005775*sin(3*ze)*sin(Q)
X!   +(0.081344 + 0.003206*nu2)*cos(ze)*sin(Q);
X! l1pert = l1pert
X    +0.015019*cos(2*ze)*sin(Q)
X    +(0.085581 + 0.002494*nu2)*sin(ze)*cos(Q)
X    +(0.025328 - 0.003117*nu2)*cos(ze)*cos(Q);
X***************
X*** 663,669 ****
X    +0.014394*cos(2*ze)*cos(Q)
X    +0.006319*cos(3*ze)*cos(Q)
X    +0.006369*sin(ze)*sin(2*Q)
X!   +0.009156*sin(2*ze)*sin(2*Q)
X    +0.007525*sin(3*psi)*sin(2*Q)
X    -0.005236*cos(ze)*cos(2*Q)
X    -0.007736*cos(2*ze)*cos(2*Q)
X--- 692,699 ----
X    +0.014394*cos(2*ze)*cos(Q)
X    +0.006319*cos(3*ze)*cos(Q)
X    +0.006369*sin(ze)*sin(2*Q)
X!   +0.009156*sin(2*ze)*sin(2*Q);
X! l1pert = l1pert
X    +0.007525*sin(3*psi)*sin(2*Q)
X    -0.005236*cos(ze)*cos(2*Q)
X    -0.007736*cos(2*ze)*cos(2*Q)
X***************
X*** 673,679 ****
X    +(.0013381 + .0001226*nu2 -.0000253*nu2*nu2)*cos(V)
X    +(.0000248 - .0000121*nu2)*sin(2*V)
X    -(.0000305 + .0000091*nu2)*cos(2*V)
X!   +.0000412*sin(2*ze)
X    +.0012415*sin(Q)
X    +(.0000390 -.0000617*nu2)*sin(ze)*sin(Q)
X    +(.0000165 - .0000204*nu2)*sin(2*ze)*sin(Q)
X--- 703,710 ----
X    +(.0013381 + .0001226*nu2 -.0000253*nu2*nu2)*cos(V)
X    +(.0000248 - .0000121*nu2)*sin(2*V)
X    -(.0000305 + .0000091*nu2)*cos(2*V)
X!   +.0000412*sin(2*ze);
X! epert = epert
X    +.0012415*sin(Q)
X    +(.0000390 -.0000617*nu2)*sin(ze)*sin(Q)
X    +(.0000165 - .0000204*nu2)*sin(2*ze)*sin(Q)
X***************
X*** 684,690 ****
X    -.0000821*cos(4*ze)*sin(Q)
X    -.0000377*cos(5*ze)*sin(Q)
X    +.0000497*cos(2*psi)*sin(Q)
X!   +(.0000163 - .0000611*nu2)*cos(Q)
```

```
X    -.0012696*sin(ze)*cos(Q)
X    -.0004200*sin(2*ze)*cos(Q)
X    -.0001503*sin(3*ze)*cos(Q)
X--- 715,722 ----
X    -.0000821*cos(4*ze)*sin(Q)
X    -.0000377*cos(5*ze)*sin(Q)
X    +.0000497*cos(2*psi)*sin(Q)
X!   +(.0000163 - .0000611*nu2)*cos(Q);
X! epert = epert
X    -.0012696*sin(ze)*cos(Q)
X    -.0004200*sin(2*ze)*cos(Q)
X    -.0001503*sin(3*ze)*cos(Q)
X***************
X*** 695,701 ****
X    +(-.0000086 + .0000230*nu2)*cos(2*ze)*cos(Q)
X    +.0000461*sin(2*psi)*cos(Q)
X    -.0000350*sin(2*Q)
X!   +(.0002211 - .0000286*nu2)*sin(ze)*sin(2*Q)
X    -.0002208*sin(2*ze)*sin(2*Q)
X    -.0000568*sin(3*ze)*sin(2*Q)
X    -.0000346*sin(4*ze)*sin(2*Q)
X--- 727,734 ----
X    +(-.0000086 + .0000230*nu2)*cos(2*ze)*cos(Q)
X    +.0000461*sin(2*psi)*cos(Q)
X    -.0000350*sin(2*Q)
X!   +(.0002211 - .0000286*nu2)*sin(ze)*sin(2*Q);
X! epert = epert
X    -.0002208*sin(2*ze)*sin(2*Q)
X    -.0000568*sin(3*ze)*sin(2*Q)
X    -.0000346*sin(4*ze)*sin(2*Q)
X***************
X*** 707,713 ****
X    +.0000467*cos(3*psi)*sin(2*Q)
X    -.0000490*cos(2*Q)
X    -(.0002842 + .0000279*nu2)*sin(ze)*cos(2*Q)
X!   +(.0000128 + .0000226*nu2)*sin(2*ze)*cos(2*Q)
X    +.0000224*sin(3*ze)*cos(2*Q)
X    +(-.0001594 + .0000282*nu2)*cos(ze)*cos(2*Q)
X    +(.0002162 - .0000207*nu2)*cos(2*ze)*cos(2*Q)
X--- 740,747 ----
X    +.0000467*cos(3*psi)*sin(2*Q)
X    -.0000490*cos(2*Q)
X    -(.0002842 + .0000279*nu2)*sin(ze)*cos(2*Q)
X!   +(.0000128 + .0000226*nu2)*sin(2*ze)*cos(2*Q);
X! epert = epert
X    +.0000224*sin(3*ze)*cos(2*Q)
X    +(-.0001594 + .0000282*nu2)*cos(ze)*cos(2*Q)
X    +(.0002162 - .0000207*nu2)*cos(2*ze)*cos(2*Q)
X***************
X*** 716,722 ****
X    +.0000343*cos(4*ze)*cos(2*Q)
X    +.0000469*sin(3*psi)*cos(2*Q)
X    -.0000242*cos(3*psi)*cos(2*Q)
```

```
X!   -.0000205*sin(ze)*sin(3*Q)
X    +.0000262*sin(3*ze)*sin(3*Q)
X    +.0000208*cos(ze)*cos(3*Q)
X    -.0000271*cos(3*ze)*cos(3*Q)
X--- 750,757 ----
X    +.0000343*cos(4*ze)*cos(2*Q)
X    +.0000469*sin(3*psi)*cos(2*Q)
X    -.0000242*cos(3*psi)*cos(2*Q)
X!   -.0000205*sin(ze)*sin(3*Q);
X! epert = epert
X    +.0000262*sin(3*ze)*sin(3*Q)
X    +.0000208*cos(ze)*cos(3*Q)
X    -.0000271*cos(3*ze)*cos(3*Q)
X***************
X*** 728,734 ****
X    -0.007075*sin(ze)
X    -0.075825*sin(ze)*sin(Q)
X    -0.024839*sin(2*ze)*sin(Q)
X!   -0.008631*sin(3*ze)*sin(Q)
X    -0.072586*cos(Q)
X    -0.150383*cos(ze)*cos(Q)
X    +0.026897*cos(2*ze)*cos(Q)
X--- 763,770 ----
X    -0.007075*sin(ze)
X    -0.075825*sin(ze)*sin(Q)
X    -0.024839*sin(2*ze)*sin(Q)
X!   -0.008631*sin(3*ze)*sin(Q);
X! w1pert = w1pert
X    -0.072586*cos(Q)
X    -0.150383*cos(ze)*cos(Q)
X    +0.026897*cos(2*ze)*cos(Q)
X***************
X*** 737,743 ****
X    -(0.013597 +0.001719*nu2)*sin(ze)*sin(2*Q)
X    +(-0.007742 + 0.001517*nu2)*cos(ze)*sin(2*Q)
X    +(0.013586 - 0.001375*nu2)*cos(2*ze)*sin(2*Q)
X!   +(-0.013667 + 0.001239*nu2)*sin(ze)*cos(2*Q)
X    +0.011981*sin(2*ze)*cos(2*Q)
X    +(0.014861 + 0.001136*nu2)*cos(ze)*cos(2*Q)
X    -(0.013064 + 0.001628*nu2)*cos(2*ze)*cos(2*Q);
X--- 773,780 ----
X    -(0.013597 +0.001719*nu2)*sin(ze)*sin(2*Q)
X    +(-0.007742 + 0.001517*nu2)*cos(ze)*sin(2*Q)
X    +(0.013586 - 0.001375*nu2)*cos(2*ze)*sin(2*Q)
X!   +(-0.013667 + 0.001239*nu2)*sin(ze)*cos(2*Q);
X! w1pert = w1pert
X    +0.011981*sin(2*ze)*cos(2*Q)
X    +(0.014861 + 0.001136*nu2)*cos(ze)*cos(2*Q)
X    -(0.013064 + 0.001628*nu2)*cos(2*ze)*cos(2*Q);
X***************
X*** 751,757 ****
X  apert = .000572*sin(V) -.001590*sin(2*ze)*cos(Q)
X    +.002933*cos(V) -.000647*sin(3*ze)*cos(Q)
```

```
X    +.033629*cos(ze) -.000344*sin(4*ze)*cos(Q)
X!   -.003081*cos(2*ze) +.002885*cos(ze)*cos(Q)
X    -.001423*cos(3*ze) +(.002172 + .000102*nu2)*cos(2*ze)*cos(Q)
X    -.000671*cos(4*ze) +.000296*cos(3*ze)*cos(Q)
X    -.000320*cos(5*ze) -.000267*sin(2*ze)*sin(2*Q);
X--- 788,795 ----
X  apert = .000572*sin(V) -.001590*sin(2*ze)*cos(Q)
X    +.002933*cos(V) -.000647*sin(3*ze)*cos(Q)
X    +.033629*cos(ze) -.000344*sin(4*ze)*cos(Q)
X!   -.003081*cos(2*ze) +.002885*cos(ze)*cos(Q);
X! apert = apert
X    -.001423*cos(3*ze) +(.002172 + .000102*nu2)*cos(2*ze)*cos(Q)
X    -.000671*cos(4*ze) +.000296*cos(3*ze)*cos(Q)
X    -.000320*cos(5*ze) -.000267*sin(2*ze)*sin(2*Q);
X***************
X*** 763,769 ****
X    -.000393*sin(3*ze)*sin(Q)
X    -.000228*sin(4*ze)*sin(Q)
X    +.002138*cos(ze)*sin(Q)
X!   -.000999*cos(2*ze)*sin(Q)
X    -.000642*cos(3*ze)*sin(Q)
X    -.000325*cos(4*ze)*sin(Q)
X    -.000890*cos(Q)
X--- 801,808 ----
X    -.000393*sin(3*ze)*sin(Q)
X    -.000228*sin(4*ze)*sin(Q)
X    +.002138*cos(ze)*sin(Q)
X!   -.000999*cos(2*ze)*sin(Q);
X! apert = apert
X    -.000642*cos(3*ze)*sin(Q)
X    -.000325*cos(4*ze)*sin(Q)
X    -.000890*cos(Q)
X***************
X*** 772,778 ****
X    -.000856*sin(ze)*cos(2*Q)
X    +.000441*sin(2*ze)*cos(2*Q)
X    +.000296*cos(2*ze)*cos(2*Q)
X!   +.000211*cos(3*ze)*cos(2*Q)
X    -.000427*sin(ze)*sin(3*Q)
X    +.000398*sin(3*ze)*sin(3*Q)
X    +.000344*cos(ze)*cos(3*Q)
X--- 811,818 ----
X    -.000856*sin(ze)*cos(2*Q)
X    +.000441*sin(2*ze)*cos(2*Q)
X    +.000296*cos(2*ze)*cos(2*Q)
X!   +.000211*cos(3*ze)*cos(2*Q);
X! apert = apert
X    -.000427*sin(ze)*sin(3*Q)
X    +.000398*sin(3*ze)*sin(3*Q)
X    +.000344*cos(ze)*cos(3*Q)
X***************
X*** 1002,1009 ****
X    +.001051*cos(2*th);
```

```
X  trans(r,b,ll,Stheta,Sr,epli, MAGNEP, "PN", "Neptune");
X
X  putchar('\n');
X! close(logfile);
X  exit(0);
X  } /* end of program main */
X
X--- 1042,1061 ----
X    +.001051*cos(2*th);
X  trans(r,b,ll,Stheta,Sr,epli, MAGNEP, "PN", "Neptune");
X
X+ #ifdef ST_MWC
X+ Cconws("\r\n");
X+ #else /* !ST_MWC */
X  putchar('\n');
X! #endif /* ST_MWC */
X! fclose(logfile);
X! #ifdef ST_MWC /* We want to hold the screen if invoked from the GEM desktop; */
X!             /* the desktop doesn't usually set any environment variables, */
X!             /* and if getenv() does find anything, it's probably empty */
X! if (((getenv("PATH")) == 0) || (strlen(getenv("PATH")) == 0)) {
X!      Cconws("press any key to continue: "); /* Hold screen if desktop */
X!      i = Crawcin(); /* Read raw character input */
SHAR_EOF
echo "End of part 1, continue with part 2"
echo "2" > s2_seq_.tmp
exit 0


From 3194966902178911232
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,9606970957a66871,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-01-19 00:59:13 PST
Path: utzoo!attcan!uunet!lll-winken!ncis.llnl.gov!ncis!helios.ee.lbl.gov!pasteur!
ucbvax!agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i093: starchrt -- Diffs to Unix starchart program for ST  part02/02
Keywords: shar, diff, C
Message-ID: <6028@saturn.ucsc.edu>
Date: 19 Jan 89 08:59:13 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1013
Approved: koreth@ssyx.ucsc.edu

Submitted-by: cs.buffalo.edu!sigmast!dgy
Posting-number: Volume 1, Issue 93
Archive-name: starchrt/part02

#!/bin/sh
# this is part 2 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
```

```
# file planet.dif continued
#
CurArch=2
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> planet.dif
X! }
X! #endif /* ST_MWC */
X  exit(0);
X  } /* end of program main */
X
X***************
X*** 1023,1031 ****
X--- 1075,1089 ----
X  double aint(z)
X    double z;
X  {
X+ #ifdef ST_MWC /* In Mark Williams C ints are 16 bits long, so use longs */
X+   long trunk;
X+
X+   trunk = (long)z;
X+ #else /* !ST_MWC */
X    int trunk;
X
X    trunk = (int)z;
X+ #endif /* ST_MWC */
X    z = (double) trunk;
X    return(z);
X  }
SHAR_EOF
chmod 0600 planet.dif || echo "restore of planet.dif fails"
sed 's/^X//' << 'SHAR_EOF' > staranim.c &&
X#include <stdio.h>
X#include <osbind.h> /* Needed for Cconws() and Crawcin() */
X
Xextern char *getenv();
X
Xint daysinmonth[12] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };
X
Xmain(argc, argv)
Xint argc;
Xchar *argv[];
X{
X     int debug = 0, i, j, month;
X     char *eargv[20], *eenv[2], label[10], picture[14], sday[2], smonth[3];
X
```

```
X     if (argc < 4) {
X          Cconws("Usage: ");
X          Cconws(argv[0]);
X          Cconws(" <month> <year> <arguments ... >\r\n");
X          Cconws("       <month> is an integer from 1 to 12.\r\n");
X          Cconws("       <year> is a four-digit integer.\r\n");
X          Cconws("       <arguments> are passed to starst.\r\n");
X          Cconws("Portions of this program, copyright 1984, ");
X          Cconws("Mark Williams Company.\r\n");
X          /* We want to hold the screen if invoked from the GEM desktop;
X             the desktop doesn't usually set any environment variables,
X             and if getenv() does find anything, it's probably a NULL string. */
X          if (((getenv("PATH")) == 0) || (strlen(getenv("PATH")) == 0)) {
X               Cconws("press any key to continue:  ");
X               i = Crawcin(); /* Read raw character input */
X          }
X          exit(1);
X     }
X     if (strlen(argv[2]) != 4) {
X          Cconws("Year must be in 4-digit form (e.g. 1988).\r\n");
X          if (((getenv("PATH")) == 0) || (strlen(getenv("PATH")) == 0)) {
X               Cconws("press any key to continue:  ");
X               i = Crawcin(); /* Read raw character input */
X          }
X          exit(1);
X     }
X     if ((month = atoi(argv[1])) > 100) {
X          month -= 100;
X          ++debug;
X     }
X     if ((month < 1) || (month > 12)) {
X          Cconws("Month must be an integer from 1 to 12.\r\n");
X          if (((getenv("PATH")) == 0) || (strlen(getenv("PATH")) == 0)) {
X               Cconws("press any key to continue:  ");
X               i = Crawcin(); /* Read raw character input */
X          }
X          exit(1);
X     }
X     for (i = 1; i <= daysinmonth[month - 1]; i++) {
X          sprintf(smonth, "%d", month);
X          sprintf(sday, "%d", i);
X          eargv[0]  = "planet.ttp";
X          eargv[1]  = "-y";
X          eargv[2]  = argv[2];
X          eargv[3]  = "-m";
X          eargv[4]  = smonth;
X          eargv[5]  = "-d";
X          eargv[6]  = sday;
X          eargv[7]  = "-t";
X          eargv[8]  = "12";
X          eargv[9]  = "-z";
X          eargv[10] = "0";
X          eargv[11] = eenv[1] = 0;
```

```
X          eenv[0] = "PATH=."; /* Make programs think they came from shell */
X          if (debug) {
X               Cconws("\r\n");
X               for (j = 0; eargv[j] != '\0'; j++) {
X                    Cconws(eargv[j]);
X                    Cconws(" ");
X               }
X          } else {
X               execve(eargv[0], eargv, eenv);
X          }
X          sprintf(label, "%02d/%02d/%2s", month, i, argv[2] + 2);
X          sprintf(picture, "star%02d%02d.pi1", month, i);
X          eargv[0]  = "starst.ttp";
X          eargv[1]  = "-t";
X          eargv[2]  = label;
X          eargv[3]  = "-x";
X          eargv[4]  = picture;
X          for (j = 3; j < argc; j++) {
X               eargv[j + 2] = argv[j];
X          }
X          eargv[j + 2] = '\0';
X          if (debug) {
X               Cconws("\r\n");
X               for (j = 0; eargv[j] != '\0'; j++) {
X                    Cconws(eargv[j]);
X                    Cconws(" ");
X               }
X               Cconws("\r\n");
X          } else {
X               execve(eargv[0], eargv, eenv);
X          }
X     } /* End of "for" loop */
X     if (debug && ((getenv("PATH")) == 0) || (strlen(getenv("PATH")) == 0)) {
X          Cconws("press any key to continue:  ");
X          i = Crawcin(); /* Read raw character input */
X     }
X}
SHAR_EOF
chmod 0600 staranim.c || echo "restore of staranim.c fails"
sed 's/^X//' << 'SHAR_EOF' > starargs.c &&
X/*
X** Written by Dave Yearke (dgy@sigmast), September 1988.
X** Portions of this program (c) Mark Williams Company.
X*/
X
X#include <osbind.h> /* Needed for Cconws() and Crawcin() */
X
Xmain()
X{
X     char *argptr, command[130], *eargv[20], *eenv[2];
X     int i;
X
X     Cconws("STARARGS - Utility for running Starchart ");
```

```
X     Cconws("software from the desktop.\r\n");
X     Cconws("Portions of this program, copyright 1984, ");
X     Cconws("Mark Williams Company.");
X     for (;;) {
X          *command = 128; /* Number of characters Cconrs() should read */
X          Cursconf(1, 0); /* Show cursor */
X          Cconws("\r\n\nPlease enter the full name of the program to be run ");
X          Cconws("(e.g. starst.ttp)\r\n");
X          Cconws("and its arguments, or an empty line to quit:\r\n");
X          Cconrs(command); /* Read an edited string from standard input */
X          if (*(command + 1) == 0) /* Number of chars actually read */
X               exit();
X          Cconws("\r\n");
X          command[*(command + 1) + 2] = '\0'; /* Make sure it's terminated */
X          i = 0;
X          argptr = command + 2;
X          do {
X               eargv[i] = argptr;
X               while ((*++argptr != ' ') && (*argptr != '\0'))
X                    ; /* Browse until space or end of string */
X               if (*argptr == ' ') /* If we found a space */
X                    *argptr++ = '\0'; /* End the string preceding this point */
X               while ((*argptr == ' ') && (*argptr != '\0'))
X                    ++argptr; /* Browse until non-space or end of string */
X               Cconws(eargv[i++]);
X               Cconws(" ");
X          } while (*argptr != '\0');
X          eargv[i] = eenv[1] = 0;
X          eenv[0] = "PATH="; /* The programs will think they're being run */
X          Cconws("\r\n");     /* from a shell. */
X          execve(eargv[0], eargv, eenv);
X     }
X}
SHAR_EOF
chmod 0600 starargs.c || echo "restore of starargs.c fails"
sed 's/^X//' << 'SHAR_EOF' > starchrt.dif &&
X*** ../starchart/starchart.c    Mon Sep 19 09:12:20 1988
X--- starchrt.c    Mon Sep 26 23:50:39 1988
X***************
X*** 58,72 ****
X--- 58,87 ----
X   ![11] the -g flag sets a mag limit for star proper names, else use codes
X   */
X
X+ #ifdef ST_MWC /* Mark Williams C for the Atari ST */
X+ #define SYSV /* MWC is almost SYSV compatible */
X+ #endif /* ST_MWC */
X+
X  #include <stdio.h>
X  #include <math.h>
X  #ifndef SYSV
X  #include <strings.h>
X  #else
```

```
X+ #define index strchr
X+ #ifndef SYSVR3 /* System V release 3 doesn't have this header file */
X+ #ifndef ST_MWC
X  #include <string.h>
X+ #else /* ST_MWC */
X+ char *strchr();
X+ #endif /* ST_MWC */
X+ #endif /* SYSVR3 */
X  #endif
X  #include <ctype.h>
X+ #ifndef ST_MWC
X  #include "starchart.h"
X+ #else /* ST_MWC */
X+ #include "starchrt.h"
X+ #endif /* ST_MWC */
X
X  /*
X   * default datasets are local, unless defined in Makefile
X***************
X*** 122,127 ****
X--- 137,149 ----
X  int annodetail, objdetail;    /* flags to control output */
X  int annoswitch, objswitch;    /* switches as seen on command line */
X  float olat, olon;        /* save areas for track clipping */

X+ #ifdef ST_MWC
X+ #ifdef INTERACTIVE
X+ extern int oldrez; /* Used to see if we've initialized the screen. */
X+ int exit_and_save = FALSE; /* Don't wait for keypress when done, save file */
X+ char picturefile[15]; /* into "picturefile" (def = star.pi1) and exit */
X+ #endif /* INTERACTIVE */
X+ #endif /* ST_MWC */
X
X  /* the code */
X
X***************
X*** 253,258 ****
X--- 275,297 ----
X         case 'b': bigflag = TRUE; break;
X         case 'a': annoswitch = TRUE; break;
X         case 'o': objswitch = TRUE; break;
X+ #ifdef ST_MWC
X+ #ifdef INTERACTIVE
X+                 /* Exit immediately, saving the picture in a DEGAS or
X+                    NeoChrome file (Chosen by .pi1 or .neo extender */
X+        case 'x': exit_and_save = TRUE;
X+                  if (((j+1) < argc) && (argv[j+1][0] != '-'))
X+                                strcpy(picturefile, argv[++j]);
X+                           else
X+                                strcpy(picturefile, "star.pi1");
X+                           break;
X+ #else /* !INTERACTIVE */
X+                 /* Help for the GEM Desktop, which cannot redirect. */
```

```
X+       case 'x': if (((j+1) < argc) && (argv[j+1][0] != '-'))
X+                           freopen(argv[++j], "wb", stdout);
X+                           break;
X+ #endif /* INTERACTIVE */
X+ #endif /* ST_MWC */
X        default:  die("unknown switch - %s", argv[j]); break;
X        }
X       if (j == argc) die("trailing command line flag - %s", argv[j-1]);
X***************
X*** 373,379 ****
X--- 412,426 ----
X      if (!title) title = "LEGEND";
X      rastr(ras, chart->racen);
X      declstr(dls, chart->dlcen);
X+ #ifdef ST_MWC
X+ #ifdef INTERACTIVE /* This bleeds over into the thumbnail, so trim it */
X+     sprintf(outstr, "(%s,%s)", ras, dls, chart->maglim);
X+ #else /* !INTERACTIVE */
X      sprintf(outstr, "(%s,%s lim: %2.1f)", ras, dls, chart->maglim);
X+ #endif /* INTERACTIVE */
X+ #else /* !ST_MWC */
X+     sprintf(outstr, "(%s,%s lim: %2.1f)", ras, dls, chart->maglim);
X+ #endif /* ST_MWC */
X  /*
X   * there are reports that large point sizes (eg "16", below) cause characters
X   * to overlap on some (pic?) output devices. To fix, set values to "10".
X***************
X*** 417,422 ****
X--- 464,481 ----
X
X      pvecsize(10); pvecsyms( 95,75,"double");   pdrawStar( 65,75, 2, 'D', NULL);
X      pvecsize(10); pvecsyms(260,75,"variable"); pdrawStar(230,75, 2, 'V', NULL);
X+ #ifdef ST_MWC
X+ #ifdef INTERACTIVE /* Adjust the legend so symbols don't overlap labels */
X+     pvecsize(10); pvecsyms( 95,50,"planet");   pdrawPlan( 75,50, 1, 'S', NULL);
X+                               pdrawPlan( 55,50, 1, 's', NULL);
X+                               pdrawPlan( 35,50, 1, 'V', NULL);
X+     pvecsize(10); pvecsyms(260,50,"galaxy");   pdrawGalx(240,50, 1, 'E', NULL);
X+                               pdrawGalx(215,50, 1, 'S', NULL);
X+     pvecsize(10); pvecsyms( 95,25,"nebula");   pdrawNebu( 75,25, 1, 'D', NULL);
X+                               pdrawNebu( 45,25, 1, 'P', NULL);
X+     pvecsize(10); pvecsyms(260,25,"cluster");  pdrawClus(240,25, 1, 'O', NULL);
X+                               pdrawClus(215,25, 1, 'G', NULL);
X+ #else /* !INTERACTIVE */
X      pvecsize(10); pvecsyms( 95,50,"planet");   pdrawPlan( 65,50, 1, 'S', NULL);
X      pvecsize(10); pvecsyms(260,50,"galaxy");   pdrawGalx(230,50, 1, 'E', NULL);
X                               pdrawGalx(205,50, 1, 'S', NULL);
X***************
X*** 424,429 ****
X--- 483,490 ----
X                               pdrawNebu( 40,25, 1, 'P', NULL);
X      pvecsize(10); pvecsyms(260,25,"cluster");  pdrawClus(230,25, 1, 'O', NULL);
X                               pdrawClus(205,25, 1, 'G', NULL);
```

```
X+ #endif /* INTERACTIVE */
X+ #endif /* ST_MWC */
X      }
X
X  chartbanner(chart)
X***************
X*** 495,501 ****
X    if (sbuf[11] != 0)
X    code[0] = sbuf[15];
X    subcode[0] = sbuf[16];
X!   color [0] = '\0'; strcpy (color, "  "); /* set unknowns to blanks */
X    label [0] = '\0';
X    name [0] = '\0'; strncat (name, &sbuf[17], strlen (&sbuf [17]) - 1);
X    }
X--- 556,562 ----
X    if (sbuf[11] != 0)
X    code[0] = sbuf[15];
X    subcode[0] = sbuf[16];
X!   strcpy(color, "  "); /* set unknowns to blanks */
X    label [0] = '\0';
X    name [0] = '\0'; strncat (name, &sbuf[17], strlen (&sbuf [17]) - 1);
X    }
X***************
X*** 554,566 ****
X--- 615,635 ----
X  /*
X   * extract color, label and name
X   */
X+ #ifdef ST_MWC
X+                 len = strlen(sbuf);
X+                 if (len > 16)
X+ #endif /* ST_MWC */
X        strncat (color, &sbuf[16], 2);
X+ #ifdef ST_MWC
X+                 if (len > 18)
X+ #endif /* ST_MWC */
X        strncat (label, &sbuf[18], strlen (&sbuf [18]) - 1);
X        if  ((len = strlen (label)) > 5)
X             {
X             strncat (name, &label[5], len - 5);
X             label [5] = '\0';
X             }
X+                 break;
X       }
X    }
X      return(0);
X***************
X*** 896,901 ****
X--- 965,971 ----
X      return (new -= b);
X      }
X
X+ #ifndef INTERACTIVE
```

```
X  die(a,b)
X      char *a, *b;
X      {
X***************
X*** 904,909 ****
X--- 974,980 ----
X      fprintf(stderr,"\n");
X      exit(1);
X      }
X+ #endif /* !INTERACTIVE */
X
X  #define LINELEN 80
X  static char legend[LINELEN];
SHAR_EOF
chmod 0600 starchrt.dif || echo "restore of starchrt.dif fails"
sed 's/^X//' << 'SHAR_EOF' > starlase.dif &&
X*** ../starchart/starlaser.c    Mon Sep 19 09:11:19 1988
X--- starlase.c    Mon Sep 26 23:41:46 1988
X***************
X*** 25,31 ****
X--- 25,35 ----
X  */
X
X  #include <stdio.h>
X+ #ifndef ST_MWC
X  #include "starchart.h"
X+ #else /* ST_MWC */
X+ #include "starchrt.h"
X+ #endif /* ST_MWC */
X
X  char *calloc ();
X
SHAR_EOF
chmod 0600 starlase.dif || echo "restore of starlase.dif fails"
sed 's/^X//' << 'SHAR_EOF' > starpost.dif &&
X*** ../starchart/starpost.c     Mon Sep 19 09:11:50 1988
X--- starpost.c    Mon Sep 26 23:43:16 1988
X***************
X*** 15,21 ****
X--- 15,25 ----
X   */
X
X  #include <stdio.h>
X+ #ifndef ST_MWC
X  #include "starchart.h"
X+ #else /* ST_MWC */
X+ #include "starchrt.h"
X+ #endif /* ST_MWC */
X
X  /*
X   * Chart parameters (limiting magnitude and window x,y,w,h)
SHAR_EOF
chmod 0600 starpost.dif || echo "restore of starpost.dif fails"
```

```
sed 's/^X//' << 'SHAR_EOF' > starst.c &&
X/*
X** Atari ST driver for starchart.
X** Works on color systems only, compiled with Mark Williams C.
X** Written by Dave Yearke (dgy@sigmast), September 1988.
X** Portions of this program (c) Mark Williams Company.
X** Thanks to the authors of the other starchart drivers for giving
X** me some examples to work from.
X*/
X
Xextern int exit_and_save;
Xextern char *getenv(), picturefile[];
X
X#include <ctype.h>    /* isprint(), iscntrl(), etc. */
X#include <linea.h>    /* Line A (low-level graphics) routines. */
X#include <osbind.h>   /* Operating system bindings */
X#include <vdibind.h>  /* The virtual device interface routines */
X#include <xbios.h>    /* Extended BIOS bindings */
X#include "starchrt.h" /* Starchart information */
X
X/* Global line A variables used by vdi; MUST be included */
Xint contrl[12], intin[128], ptsin[128], intout[128], ptsout[128];
X/* Array used by vs_clip() */
Xint cliparray[] = { 1, 1, 319, 199 };
X/* Arrays used by v_opvwk() */
Xint work_in[] = { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2 };
Xint work_out[57];
X
X#define TRUE 1
X#define NULL 0
X
X/*
X** Starchart was designed for 1024x768 pixels.  These macros scale the
X** image to the 320x200 Atari ST low-resolution (16-color) mode.  Also,
X** the y coord is subtracted from 200 so the image isn't upside-down.
X*/
X#define xadjust(x) ((x) * 0.3125)
X#define yadjust(y) (200 - ((y) * 0.2604167))
X
Xstruct la_font *fontp; /* Line A font header. */
Xchar line[100], linemode, *p;
Xchar scr_wrk[1024];
Xint vdihandle; /* Virtual device's handle */
X
Xint CURRX, CURRY, currez, old_color[16], oldrez = 99; /* 99 is impossible! */
X
X/*
X** Settings for the palette, given as RGB, 3 bits for each color (512 total).
X** I tried to use different registers for different types of objects so they
X** could be set individually (for example, in DEGAS you could blink a register
X** for an object you're trying to point out).
X*/
Xint new_color[16] = { 0x000,   /* Black      - Background color */
```

```
X                       0x777,   /* White      - 1st and higher mag. stars */
X                       0x666,   /* White      - 2nd mag. stars */
X                       0x555,   /* White      - 3rd mag. stars */
X                       0x444,   /* White      - 4th mag. stars */
X                       0x333,   /* White      - 5th mag. stars */
X                       0x222,   /* White      - 6th and lower mag. stars */
X                       0x222,   /* White      - Nebulae, Galaxies, Clusters */
X                       0x007,   /* Blue       - 1st and higher mag. stars */
X                       0x700,   /* Red        - 1st and higher mag. stars */
X                       0x770,   /* Yellow     - Sol */
X                       0x704,   /* Pink       - Inferior Planets */
X                       0x404,   /* Purple     - Superior Planets */
X                       0x020,   /* Dark Green - Dotted lines */
X                       0x200,   /* Dark Red   - Hyphenated lines */
X                       0x003 }; /* Deep Blue  - Text and borders */
X
X/*
X** These are reverse-image colors for the background and stars, used when the
X** 'r' key is hit when the program pauses before exiting.  Switching these
X** registers makes a nice output image for a screen dump to a printer.
X*/
Xint rev_color[8]  = { 0x777,   /* White      - Background color */
X                       0x000,   /* Black      - 1st and higher mag. stars */
X                       0x222,   /* Black      - 2nd mag. stars */
X                       0x333,   /* Black      - 3rd mag. stars */
X                       0x444,   /* Black      - 4th mag. stars */
X                       0x555,   /* Black      - 5th mag. stars */
X                       0x666,   /* Black      - 6th and lower mag. stars */
X                       0x666 }; /* Black      - Nebulae, Galaxies, Clusters */
X
X/*
X** Chart parameters (limiting magnitude and window x,y,w,h)
X*/
X
Xmapblock thumbnail =  { 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
X             3.2, 1.0, 2.05, 420, 35, 480, 195, 0.0 };
X
Xmapblock master = { 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
X             8.0, 2.0, 2.05, 20, 265, 880, 500, 0.0 };
X
Xmapblock bigmaster =  { 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
X             8.0, 2.5, 2.5, 20,  65, 880, 700, 0.0 };
X
X/*
X** Generic functions
X*/
X
X/*
X** Initialize the graphics mode
X*/
Xvecopen()
X{
X     int i;
```

```
X
X     Cconws("\033H\033J"); /* Clear the screen */
X     if ((oldrez = Getrez()) == GR_HIGH) {
X          die("%s\n", "Sorry, this works on color systems only.");
X          Pterm(1);
X     } else if (oldrez == GR_MED)
X          Setscreen((char *)-1, (char *)-1, GR_LOW); /* Set low resolution */
X     currez = GR_LOW;
X     appl_init(); /* Initialize the application and register it with AES */
X     vdihandle = graf_handle(&i, &i, &i, &i); /* Get a VDI handle */
X     v_opnvwk(work_in, &vdihandle, work_out); /* Open the virtual workscreen */
X     vs_clip(vdihandle, 1, cliparray); /* Set the clipping rectangle */
X     for (i = 0; i < 16; i++) /* Save and set the palette */
X          old_color[i] = Setcolor(i, new_color[i]);
X
X     linea0(); /* Initialize the line A routines */
X     lineaa(); /* Hide mouse pointer */
X     Cursconf(0, 0); /* Hide cursor */
X
X     WMODE = 0; /* Writing mode = replace */
X     STYLE = 0; /* Normal Letters */
X     SRCY = 0; /* Y coord of character in font */
X     LITEMSK = 0x5555; /* Lightening and skewing masks for text */
X     SKEWMSK = 0x1111;
X     TEXTFG = 15; /* Text foreground dark blue */
X     SCRTCHP = scr_wrk; /* Scratch area for graphics */
X     LNMASK = -1; /* Solid lines */
X     WEIGHT = 1; /* Thickening factor for lines */
X     LSTLIN = -1;
X}
X
X/*
X** Clean up
X*/
Xvecclose ()
X{
X     register int i;
X
X     if (exit_and_save)
X          savepic();
X     else
X          while (1) { /* Loop if 'r' is pressed, return for anything else */
X               switch (i = Crawcin()) { /* Wait for keypress before quitting */
X               case 'r': /* Reverse the color registers */
X                    if (Setcolor(0, -1) == 0) { /* If normal, reverse it */
X                         for (i = 0; i < 8; i++)
X                              Setcolor(i, rev_color[i]);
X                    } else { /* Make it normal */
X                         for (i = 0; i < 8; i++)
X                              Setcolor(i, new_color[i]);
X                    }
X                    continue; /* Restart the loop */
X                    break;
```

```
X               case 's': /* Save as a DEGAS picture */
X                    strcpy(picturefile, "star.pi1");
X                    savepic();
X                    break;
X               case 'S': /* Save as a NeoChrome picture */
X                    strcpy(picturefile, "star.neo");
X                    savepic();
X                    break;
X               default:
X                    break;
X               }
X               break;
X          }
X     vecreset();
X}
X
X/*
X** Restore the screen
X*/
Xvecreset()
X{
X     int i;
X
X     v_clsvwk(vdihandle); /* Close the virtual workscreen */
X     appl_exit(); /* Remove the application from AES */
X     Cconws("\033H\033J"); /* Clear the screen */
X     Setscreen((char *)-1, (char *)-1, oldrez); /* Set resolution */
X     for (i = 0; i < 16; i++) /* Set the colors back to their saved values */
X          Setcolor(i, old_color[i]);
X     if (((getenv("PATH")) == 0) || (strlen(getenv("PATH")) == 0))
X          linea9(); /* Show mouse pointer */
X     else /* If called from a shell */
X          Cursconf(1, 0); /* Show cursor */
X}
X
X/*
X** Save the screen as a DEGAS or NeoChrome picture
X*/
Xsavepic()
X{
X     int neo = 0, file; /* Assume DEGAS file */
X
X     if (strrchr(picturefile, '.') != NULL) { /* Extender given */
X          if ((!strncmp(picturefile + strrchr(picturefile, '.'), ".neo", 4)) ||
X             (!strncmp(picturefile + strrchr(picturefile, '.'), ".NEO", 4)))
X               ++neo; /* Save as a NeoChrome picture */
X     } else
X          strcat(picturefile, ".pi1");
X     if ((file = creat(picturefile, 0)) == -1)
X          die("Cannot open %s.", picturefile);
X     if (neo) /* NeoChrome has two extra null bytes at the beginning */
X          if (write(file, &currez, 2) != 2)
X               die("Write failure on file %s.", picturefile);
```

```
X     if (write(file, &currez, 2) != 2) /* Write the resolution */
X          die("Write failure on file %s.", picturefile);
X     if (write(file, &new_color[0], 32) != 32) /* Write the palette info */
X          die("Write failure on file %s.", picturefile);
X     if (neo) /* NeoChrome has a 128-byte header, so we'll pad with 94 bytes */
X          for (neo = 0; neo < 47; ++neo) /* Recycle the variable "neo" */
X               if (write(file, &currez, 2) != 2)
X                    die("Write failure on file %s.", picturefile);
X     if (write(file, Physbase(), 32000) != 32000) /* Write the screen */
X          die("Write failure on file %s.", picturefile);
X     close(file); /* Done!  (that wasn't too bad, was it?) */
X}
X
X/*
X** This function is duplicated in starchrt.c for non-ST versions, using
X** the INTERACTIVE manifest constant.
X*/
Xdie(a,b)
Xchar *a, *b;
X{
X     char buf[80];
X     int i;
X
X     if (oldrez != 99) /* See if we ever initialized the screen. */
X         vecreset(); /* Reset the screen to something sane */
X     if (!strncmp(b, "\nusage", 6)) { /* Sub in the ST's usage */
X          Cconws("\n\rusage:\tstar* [ Ra Dcl Scale Title Maglim Labellim ]\n");
X          Cconws("\ror\tstar* [ -r Ra -d Dcl -s Scale -t Title -m Maglim -l ");
X          Cconws("Labellim -f x.str ]\n\ror\tstar* [ -c con (3 or 4 letters ");
X          Cconws("chosen from con.loc) -l ... ]\n\ror\tstarst -x [ file.PI1 ");
X          Cconws("| file.NEO ] ... (to exit and save screen)\n\n\r");
X          Cconws("Portions of this program, copyright 1984, ");
X          Cconws("Mark Williams Company");
X     } else {
X          sprintf(buf, a, b);
X          Cconws(buf);
X     }
X     Cconws("\n\r"); /* Cconws does not map \n into \n\r */
X     /* We want to hold the screen if invoked from the GEM desktop;
X        the desktop doesn't usually set any environment variables,
X        and if getenv() does find anything, it's probably a NULL string. */
X     if (((getenv("PATH")) == 0) || (strlen(getenv("PATH")) == 0)) {
X          Cconws("press any key to continue:  "); /* Hold screen if desktop */
X          i = Crawcin(); /* Read raw character input */
X     }
X    exit(1);
X}
X
X/*
X** Functions for manipulating text.
X*/
X
X/*
```

```
X** This function sets the point size for text. (not applicable)
X*/
Xvecsize (points)
Xint points;
X{
X}
X
X/*
X** This function puts text on the screen.
X*/
Xvecsyms (x, y, s)
Xint x,y;
Xchar *s;
X{
X     register char *ptr;
X     register unsigned int tmp;
X
X     if (*s == '\0') /* Don't bother if the string is empty */
X          return;   /* (A null char can make linea8() crash!) */
X     fontp = la_init.li_a1[0];  /* 6x6 system font */
X     FBASE = fontp->font_data; /* Pointer to start of font form */
X     FWIDTH = fontp->font_width; /* Width of font form */
X     DELY = fontp->font_height; /* Height of character in font */
X
X     DSTX = xadjust(x);
X     if ((DSTY = yadjust(y)) > 2)
X          DSTY -= 2; /* This vertically centers the text on the line */
X     if (DSTY > 191)
X          DSTY = 191; /* Make sure the bottom line shows up */
X     ptr = s;
X     do { /* Output the characters until the end of the string is hit */
X          if (!iscntrl(*ptr)) {
X               tmp = *ptr - fontp->font_low_ade; /* Find the character data */
X               SRCX = fontp->font_char_off[tmp]; /* within the font data */
X               DELX = fontp->font_char_off[tmp + 1] - SRCX;
X               /* Check screen boundaries */
X               if ((DSTX > 0) && (DSTX < 311) && (DSTY > 0))
X                    if (isspace(*ptr)) {
X                         WMODE = 1; /* Writing mode = transparent */
X                         linea8();
X                         WMODE = 0; /* Writing mode = replace */
X                    } else
X                         linea8(); /* Line A trap for text blit routine */
X          }
X     } while (*++ptr != '\0');
X}
X
X/*
X** This function puts text on the screen using the greek alphabet.
X** (Well, not yet, but someday ... :-))
X*/
Xvecsymsgk(s, x, y)
Xchar *s;
```

```
X{
X     vecsyms(s, x, y);
X}
X
X/*
X** Functions for drawing points and lines.
X*/
X
X/*
X** Change the current position of x and y.
X*/
Xvecmove (x, y)
Xint x,y;
X{
X     CURRX = x;
X     CURRY = y;
X}
X
X/*
X** Move to a new location (x1, y1) and draw a solid line to (x2, y2).
X*/
Xvecmovedraw (x1, y1, x2, y2)
Xint x1, y1, x2, y2;
X{
X     vecmove(x1, y1);
X     vecdraw(x2, y2);
X}
X
X/*
X** Draw a solid line to (x,y)
X*/
Xvecdraw (x, y)
Xint x,y;
X{
X     if (linemode != 'S') {
X          linemode = 'S';
X          COLBIT0 = 1; /* Set the bit plane for line drawing (color 15) */
X          COLBIT1 = 1;
X          COLBIT2 = 1;
X          COLBIT3 = 1;
X     }
X     X1 = xadjust(CURRX);
X     Y1 = yadjust(CURRY);
X     X2 = xadjust(CURRX = x);
X     Y2 = yadjust(CURRY = y);
X     linea3(); /* Line blit routine */
X}
X
X/*
X** Draw a dotted line (dark green on the ST) to (x,y)
X*/
Xvecdrawdot(x, y)
X{
```

```
X     if (linemode != 'D') {
X          linemode = 'D';
X          COLBIT0 = 1; /* Set the bit plane for line drawing (color 13) */
X          COLBIT1 = 0;
X          COLBIT2 = 1;
X          COLBIT3 = 1;
X     }
X     X1 = xadjust(CURRX);
X     Y1 = yadjust(CURRY);
X     X2 = xadjust(CURRX = x);
X     Y2 = yadjust(CURRY = y);
X     linea3();
X}
X
X/*
X** Draw a hyphenated line (dark red on the ST) to (x,y)
X*/
Xvecdrawhyph(x, y)
X{
X     if (linemode != 'H') {
X          linemode = 'H';
X          COLBIT0 = 0; /* Set the bit plane for line drawing (color 14) */
X          COLBIT1 = 1;
X          COLBIT2 = 1;
X          COLBIT3 = 1;
X     }
X     X1 = xadjust(CURRX);
X     Y1 = yadjust(CURRY);
X     X2 = xadjust(CURRX = x);
X     Y2 = yadjust(CURRY = y);
X     linea3();
X}
X
X/*
X** Draw a horizontal line.
X*/
Xdrawlen (x, y, dx, dy, len)
Xint x, y, dx, dy, len;
X{
X     if (linemode != 'S') {
X          linemode = 'S';
X          COLBIT0 = 1; /* Set the bit plane for line drawing (color 15) */
X          COLBIT1 = 1;
X          COLBIT2 = 1;
X          COLBIT3 = 1;
X     }
X     X1 = xadjust(x + dx);
X     Y1 = yadjust(y + dy);
X     X2 = xadjust(CURRX = x + dx + len - 1);
X     Y2 = yadjust(CURRY = y + dy);
X     linea3();
X}
X
```

```
X/*
X** Functions for astronomical objects.
X*/
X
X/*
X** Draw the sun or a planet
X*/
XdrawPlan(x, y, mag, type, color)
Xint x, y, mag, type;
Xchar *color;
X{
X     if (type == 'S')
X          INTIN[0] = 10; /* Yellow for the Sun */
X     else if ((type == 'M') || (type == 'V'))
X          INTIN[0] = 11; /* Pink for inferior planets (Mercury or Venus) */
X     else
X          INTIN[0] = 12; /* Purple for superior planets (m, J, s, U, or N) */
X     PTSIN[0] = xadjust(CURRX = x);
X     PTSIN[1] = yadjust(CURRY = y);
X     if (linea2() == 0) /* See if there's anything there already */
X          linea1(); /* Put a pixel on the screen */
X}
X
X/*
X** Draw a star.
X*/
XdrawStar(x, y, mag, type, color)
Xint x, y, mag, type;
Xchar *color;
X{
X     if (mag > 6)
X          INTIN[0] = 6;
X     else if (mag < 1)
X          INTIN[0] = 1;
X     else
X          INTIN[0] = mag; /* No conversion necessary, direct map into palette */
X     if (mag < 2) /* Brightest stars will be in color if possible */
X          if (color != NULL)
X               if ((color[0] == 'O') || (color[0] == 'B'))
X                    INTIN[0] = 8; /* Color register 8 holds blue */
X               else if ((color[0] == 'K') || (color[0] == 'M'))
X                    INTIN[0] = 9; /* Color register 9 holds red */
X     PTSIN[0] = xadjust(CURRX = x);
X     PTSIN[1] = yadjust(CURRY = y);
X     if (mag < 1)
X          drawBigStar(x, y); /* Bright stars clobber anything under them */
X     else if (linea2() == 0) /* See if there's anything there already */
X               linea1();
X}
X
X/*
X** Draw a bright star as a cross shape.
X*/
```

```
XdrawBigStar(x, y)
Xint x, y;
X{
X     linea1(); /* x, y */
X     --PTSIN[0];
X     linea1(); /* x-1, y */
X     ++PTSIN[0];
X     --PTSIN[1];
X     linea1(); /* x, y-1 */
X     PTSIN[1] += 2;
X     linea1(); /* x, y+1 */
X     ++PTSIN[0];
X     --PTSIN[1];
X     linea1(); /* x+1, y */
X}
X
X/*
X** Draw a nebula. ('type' = 'D' for diffuse, 'P' for planetary)
X*/
XdrawNebu(x, y, mag, type, color)
Xint x, y, mag, type;
Xchar *color;
X{
X     /* This isn't as clever as cstar() in starimag.c, but it's fast */
X     INTIN[0] = 7; /* Make 'em faint, with this shape: */
X     PTSIN[0] = xadjust(CURRX = x) - 1; /*         **       */
X     PTSIN[1] = yadjust(CURRY = y);     /*        *  *      */
X     linea1(); /* x-1, y                          **       */
X     ++PTSIN[1];
X     linea1(); /* x-1, y+1 */
X     ++PTSIN[0];
X     ++PTSIN[1];
X     linea1(); /* x, y+2 */
X     PTSIN[1] -= 3;
X     linea1(); /* x, y-1 */
X     ++PTSIN[0];
X     ++PTSIN[1];
X     linea1(); /* x+1, y */
X     ++PTSIN[1];
X     linea1(); /* x+1, y+1 */
X}
X
X/*
X** Draw a galaxy. ('type' = 'P' for sphere, 'S' for spiral)
X*/
XdrawGalx(x, y, mag, type, color)
Xint x, y, mag, type;
Xchar *color;
X{
X     int i;
X
X     CURRX = x;
X     CURRY = y;
```

```
X     INTIN[0] = 7; /* Make 'em faint, with this shape: */
X     PTSIN[0] = xadjust(CURRX = x) - 2; /*        **  *   */
X     PTSIN[1] = yadjust(CURRY = y);     /*      ******    */
X     for (i = 0; i < 6; ++i) {          /*     *  **      */
X          PTSIN[0] += 1;
X          linea1(); /* (x-2)+i, y */
X     }
X     ++PTSIN[0];
X     --PTSIN[1];
X     linea1(); /* x+4, y-1 */
X     PTSIN[0] -= 3;
X     linea1(); /* x+1, y-1 */
X     --PTSIN[0];
X     linea1(); /* x, y-1 */
X     PTSIN[1] += 2;
X     linea1(); /* x, y+1 */
X     ++PTSIN[0];
X     linea1(); /* x+1, y+1 */
X     PTSIN[0] -= 4;
X     linea1(); /* x-3, y+1 */
X}
X
X/*
X** Draw a cluster. ('type' = 'G' for glob., 'O' for open, 'C' for Galactic)
X*/
XdrawClus(x, y, mag, type, color)
Xint x, y, mag, type;
Xchar *color;
X{
X     INTIN[0] = 7; /* Make 'em faint, with this shape: */
X     PTSIN[0] = xadjust(CURRX = x);     /*         * *      */
X     PTSIN[1] = yadjust(CURRY = y);     /*        * * *     */
X     linea1(); /* x, y                          * *      */
X     PTSIN[0] -= 2;
X     linea1(); /* x-2, y */
X     ++PTSIN[0];
X     --PTSIN[1];
X     linea1(); /* x-1, y-1 */
X     PTSIN[0] += 2;
X     linea1(); /* x+1, y-1 */
X     PTSIN[1] += 2;
X     linea1(); /* x+1, y+1 */
X     PTSIN[0] -= 2;
X     linea1(); /* x-1, y+1 */
X     PTSIN[0] += 3;
X     --PTSIN[1];
X     linea1(); /* x+2, y */
X}
SHAR_EOF
chmod 0600 starst.c || echo "restore of starst.c fails"
rm -f s2_seq_.tmp
echo "You have unpacked the last part"
exit 0
```

From 5867369051356291464
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,bf455ec068452a3b,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-01-19 01:14:04 PST
Path: utzoo!attcan!uunet!lll-winken!ncis.llnl.gov!ncis!helios.ee.lbl.gov!pasteur!
ucbvax!agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i094: passm -- Logic circuit assembler part01/02
Keywords: shar, Pascal
Message-ID: <6029@saturn.ucsc.edu>
Date: 19 Jan 89 09:14:04 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1036
Approved: koreth@ssyx.ucsc.edu

Submitted-by: sun.com!skywest!BRENES (Erasmo Brenes)
Posting-number: Volume 1, Issue 94
Archive-name: passm/part01

[Documentation is the first part of the sharfile, so page down a bit to read
 it. This program probably isn't for everyone. Binaries will appear in the
 binaries group. -sg]

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
# This is part 1 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
#
#    Run the following text with /bin/sh to create:
#      passm.doc
#      PASSM.PAS
#
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
sed 's/^X//' << 'SHAR_EOF' > passm.doc &&
X
X
X
X
X
X
X
X     _I_n_t_r_o_d_u_c_t_i_o_n
X
X             Passm, a PAL Assembler,  allows  the  user  to  write  boolean
X     equations  describing  the outputs of a PAL and generates a JEDEC fuse
```

X file. This assembler supports most 20 and 24 pin PALs. Currently, the
X list of parts supported are:
X
X
X

```
>module< module_name{;}   {optional_sections}
  device_type_declaration
  pin_list_section
  boolean_equations
>end<
```

The module_name is an identifier, no more than 15 characters, which must start with a letter. The set of valid characters for an identifier is : a..z, A..Z, underscore '_', and the minus sign '-'. Notice that the semicolon ending the module name is optional.

The next element within a module is the optional_section. This section is being reserved for future features, such as flags indicating what type of reduction to do with the boolean equations, include files, etc. At the present time, no optional features are enabled.

The next section is the device_type_declaration. This section informs PASSM as to what type of device is being targeted, ie. a 16r4, etc. This section syntax is as follows:

```
device_name  >device<  >'<part_type>';<
```

The device_name, an identifier, is the symbolic name with which the user may refer to the PAL. The part_type is the actual device being programmed. The following is the list of available devices:

20 Pin Pals	24 Pin Pals
p10l8, p10h8	p12l10
p12l6, p12h6	p14l8
p14l4, p14h4	p16l6
p16l2, p16h2	p18l4
p16l8	p20l2
p16r4	p20l10
p16r6	p20l8
p16r8	p20r4
	p20r6
	p20r8
	p22v10

```
X
X     Notice that the part_type must appear as shown above.
X
X             The next section in a module description is the pin list. This
X     section provides a symbolic reference between the signal names used in
X     the equations, and the device physical pins. The syntax followed is:
X
X
X          signal_name {, signal_name}  >pin< pin_numbers >;<
X
X
X                                    Page 2
X
X
X
X
X
X
X
SHAR_EOF
chmod 0600 passm.doc || echo "restore of passm.doc fails"
sed 's/^X//' << 'SHAR_EOF' > PASSM.PAS &&
X{************************************************************************}
X{* This program is a general purpose PAL assembler. You may copy and use   *}
X{* it for personal purposes. No commercial use of this program is allowed  *}
X{* without the consent of the author.                                      *}
X{* THIS IS THE Atari ST Version                                            *}
X{* (c) Copyright 1987,1988 by Erasmo Brenes.                               *}
X{************************************************************************}
Xprogram passm (input,output,source,simfile);
X const
X  linewidth = 40;
X  blank = ' ';  semicol = ';';  comment = '"';
X  maxterms = 19;        maxinputs = 22;
X  maxpins = 24;         npals   = 23;
X  maxcols = 44;         maxouts = 10;
X
X type
X  symbol =( ident, int, num, eql, quotes, semicolon, apostrophe,
X            leftbrkt, rightbrkt, device, pin, equations,module,flag,
X            lftparen,rgtparen,title,node,stype,macro,andoperator,
X            oroperator,invert,colon,ends,enable,preset,clear);
X  palsymb = ( p10l8,p12l6,p14l4,p16l2,p16l8,p16rx,p12l10,p14l8,p16l6,
X              p18l4,p20l2,p20l10,p20l8,p20rx,p22vx);
X  tkens = packed array [1..15] of char;
X  kind  = (reg, nonreg, bidir, tristate);
X  palsize = (input18, input22);
X  logic = (high, low);
X  trans1typ =
X   record
X        transfer : array[1..maxpins] of integer
X   end;
X  outtype =
```

```
X   record
X        outnumb : integer;
X        outname : tkens;
X        outkind : kind;
X        size    : palsize;
X        form    : logic;
X        matrix  : array [1..maxterms,1..maxcols] of char
X   end;
X  entrytype =
X   record
X        name : tkens;
X        pinn : integer
X   end;
X  string2 = packed array [1..4] of char;
X  filnam = packed array [1..80] of char;
X  ptermtyp = array [1..maxcols] of char;
X
X var
X  source,simfile : text;
X  token  : tkens;
X  palknds : array [1..npals] of char;
X  pals    : array [1..npals] of tkens;
X  symtable: array [1..maxpins] of entrytype;
X  outtable: array [1..11] of outtype;
X  palkind : palsymb;
X  fusetoinp,fusetopin : array [palsymb] of trans1typ;
X  paltyp  : array [1..npals] of palsymb;
X  filspc : string[80];
X  sym : symbol;
X  reserved : array [1..13] of tkens;
X  pdevice : tkens;
X  wsym : array [1..13] of symbol;
X  ptype,ch,tab : char;
X  nexout,outindex : integer;
X  nexin : integer;
X  value,i,j,pointer,iterm,totalterms : integer;
X  Abort,empty,pal16,found : boolean;
X  ar, sp : ptermtyp;
X
X procedure bgetchar (var ch:char);
X  begin
X   empty := false;
X   if eof(source)
X    then begin
X          empty := true;
X          ch := blank
X         end
X    else begin
X          if eoln(source)
X           then begin
X                 readln (source);
X                 ch := blank
X                end
```

```
X           else
X          if eof(source)
X           then begin
X                 empty := true;
X                 ch := blank
X                end
X           else begin
X                 read (source,ch);
X                 if ch = comment
X                  then begin
X                        repeat
X                        readln (source);
X                        if eof(source)
X                         then begin
X                                empty := true;  ch := blank
X                              end
X                         else read (source,ch)
X                        until (ch <> comment) or (eof(source))
X                       end
X                end
X         end
X  end; {bgetchar}
X
X procedure numbr;
X{this routine always leaves with ch containing the next character!}
X  var
X   j : integer;
X  begin
X   sym := int;
X   value := 0;  j:= 0;
X   repeat
X    value := 10*value + (ord(ch) - ord('0'));
X    bgetchar (ch);       j:= j + 1
X   until not(ch in ['0'..'9'])
X  end; {numbr}
X
X procedure gettoken;
X  var
X   i,j,k : integer;
X  begin
X   i:= 0;
X   while ((ch=blank)or(ch=tab))and(not empty) do bgetchar(ch);
X   if (ch in ['A'..'Z'])or(ch in['a'..'z'])or(ch = '-')
X    then begin
X          repeat
X           i:= i + 1;
X           token [i]:= ch;      bgetchar(ch)
X          until not((ch in ['A'..'Z'])or(ch in['a'..'z'])or(ch in ['0'..'9'])
X           or (ch='_')) or empty or (i = 15);
X          if not empty
X           then begin
X                 if (i < 15) then repeat
X                                  i:= i + 1; token[i]:= blank
```

```
X                                   until (i=15);
X                k := 0;
X                for j:=1 to 13 do
X                 if token = reserved[j]
X                  then k := j;
X                if k = 0
X                 then sym := ident
X                 else sym := wsym [k]
X               end
X         end
X    else begin
X          if (ch in ['0'..'9'])
X           then numbr
X           else case ch of
X                 '^':  begin
X                        sym := num;
X                        bgetchar (ch)
X                       end;
X                 '=':  begin
X                        sym := eql;
X                        bgetchar (ch)
X                       end;
X                 ';':  begin
X                        sym := semicolon;
X                        bgetchar (ch)
X                       end;
X                 '''': begin
X                        sym := apostrophe;
X                        bgetchar (ch)
X                       end;
X                 '`':  begin
X                        sym := apostrophe;
X                        bgetchar (ch)
X                       end;
X                 '"':  begin
X                        sym := quotes;
X                        bgetchar (ch)
X                       end;
X                 '[':  begin
X                        sym := leftbrkt;
X                        bgetchar (ch)
X                       end;
X                 ']':  begin
X                        sym := rightbrkt;
X                        bgetchar (ch)
X                       end;
X                 '(':  begin
X                        sym := lftparen;
X                        bgetchar (ch)
X                       end;
X                 ')':  begin
X                        sym := rgtparen;
X                        bgetchar (ch)
```

```
X                       end;
X                 '!':  begin
X                        sym := invert;
X                        bgetchar (ch)
X                       end;
X                 '&':  begin
X                        sym := andoperator;
X                        bgetchar (ch)
X                       end;
X                 '#':  begin
X                        sym := oroperator;
X                        bgetchar (ch)
X                       end;
X                 ':':  begin
X                        sym := colon;
X                        bgetchar (ch)
X                       end;
X                 otherwise:
X                    begin
X                     bgetchar (ch);
X                     gettoken { get next token }
X                    end
X               end
X         end
X  end; {gettoken}
X
X procedure semimodule;
X  begin
X   gettoken;
X   while sym = semicolon
X    do gettoken;
X  end;
X
X procedure search ( kind : integer);
X  var
X   i,j : integer;
X  begin
X   case kind of
X    1:   begin
X          pointer := 0;
X          for i:=1 to npals do
X           if token = pals[i]
X            then pointer := i
X         end;
X    2:  begin
X         j := pointer;
X         pointer := 0;
X         for i:=1 to 24 do
X          with symtable[i] do
X           if pinn = j
X            then pointer := i
X        end;
X    3:  begin      { search a signal name for its corresponding pin }
```

```
X         pointer := 0;  found := false;
X         for i:= 1 to maxpins do
X          with symtable[i] do
X           if token = name
X            then begin
X                  pointer := pinn; found := true
X                 end
X        end;
X    otherwise:
X        writeln ('!!! software error in search procedure')
X   end
X  end; {search}
X
X procedure start;
X  var
X   first : integer;
X  begin
X   while not(sym = equations) and (not Abort) and not(eof(source))do
X    begin
X     first := nexin + 1;
X     if sym = ident
X      then begin
X            nexin := nexin + 1;
X            symtable[nexin].name := token;
X            gettoken;
X            while sym = ident do
X             begin      { get list of identifiers }
X              nexin := nexin + 1;
X              symtable[nexin].name := token;
X              gettoken
X             end;
X            case sym of
X             device: begin
X                      nexin := first - 1;  {ignore all previous identifiers}
X                      gettoken;
X                      if sym = apostrophe
X                       then begin
X                             gettoken;
X                             search (1);
X                             if pointer = 0
X                              then begin
X                                    writeln ('** not a valid part ',token);
X                                    Abort := true
X                                   end
X                              else begin
X                                    pdevice := token;
X                                    ptype := palknds[pointer];
X                                    palkind := paltyp [pointer];
X                                    gettoken;
X                                    if sym = apostrophe
X                                     then gettoken;
X                                    if sym = semicolon
X                                     then gettoken
```

```
X                                    else Abort := true {screw the idiot***}
X                                 end
X                            end
X                     end;
X             pin:    begin
X                      gettoken; { it must be a pin number }
X                      while not(sym = int) do gettoken;
X                      repeat
X                       symtable[first].pinn := value;
X                       first := first + 1;
X                       gettoken
X                      until first > nexin;
X                      if sym = semicolon
X                       then gettoken
X                       else Abort := true       {screw the idiot ***}
X                     end;
X             otherwise:
X                     begin
X                      nexin := first - 1;
X                      while not (sym = semicolon)
X                        do gettoken;
X                      gettoken
X                     end
X            end
X           end
X    end
X  end;   {start}
X
X procedure titlemodule;
X  begin
X   gettoken;
X   if sym = apostrophe
X    then begin
X          repeat
X           gettoken
X          until sym = apostrophe;
X          gettoken;
X          if sym = semicolon
X           then begin
X                 semimodule;
X                 start
X                end
X           else start
X         end
X    else begin
X          writeln ('** illegal construct for the title section');
X          Abort := true
X         end
X  end;  {titlemodule}
X
X procedure flagmodule;
X  begin
X   gettoken;
```

```
X   if sym = apostrophe
X    then begin
X          repeat
X           gettoken
X          until sym = apostrophe;
X          gettoken;
X          case sym of
X           title : titlemodule;
X           semicolon: begin
X                        semimodule;
X                        if sym = title
X                         then titlemodule
X                         else start
X                      end;
X           otherwise:
X                start
X          end
X         end
X    else begin
X          writeln ('** illegal construct for the flag section');
X          Abort := true
X         end
X  end;  {flagmodule}
X
X procedure arguments;
X  begin
X   gettoken;
X   case sym of
X    ident : begin
X              gettoken;
X              while not(sym = rgtparen)
X               do gettoken;
X              gettoken;
X              case sym of
X               flag : flagmodule;
X               title: titlemodule;
X               semicolon: begin
X                           semimodule;
X                           if sym = flag
X                            then flagmodule
X                            else if sym = title
X                                  then titlemodule
X                                  else start
X                          end;
X              otherwise:
X                begin
X                 writeln ('** illegal path after module arguments');
X                 Abort := true
X                end
X             end
X            end;
X    rgtparen: begin
X                gettoken;
```

```
X                case sym of
X                 flag : flagmodule;
X                 title: titlemodule;
X                 semicolon: begin
X                             semimodule;
X                             if sym = flag
X                              then flagmodule
X                              else if sym = title
X                                    then titlemodule
X                                    else start
X                            end;
X                 otherwise:
X                        start
X                end
X              end;
X    otherwise:
X        begin
X         writeln ('** missing right parenthesis in dummy argument list');
X         Abort := true
X        end
X   end
X  end;   {arguments}
X
X procedure getnames;
X  begin
X   gettoken;
X   while not((sym = module))and (not empty)
X    do gettoken;
X   gettoken;
X   if sym = ident
X    then begin
X          gettoken;
X          case sym of
X           lftparen :   arguments;
X           flag:        flagmodule;
X           title:       titlemodule;
X           semicolon:   begin
X                         semimodule;
X                         case sym of
X                          flag : flagmodule;
X                          title: titlemodule;
X                          otherwise:
X                            start
X                         end
X                        end;
X           otherwise:
X             start
X          end
X        end
X    else begin
X          Abort := true;
X          writeln ('** missing module name')
X        end
```

```
X  end; {getnames}
X
X procedure error (errnmbr : integer);
X  begin
X   case errnmbr of
X    1 : begin
X         writeln ('Signal name undefined: ',token)
X        end;
X    2 : begin
X         writeln ('error in andoperator!')
X        end;
X    3 : begin
X         writeln ('Expecting a signal name');
X         writeln ('Undetermined token ',token)
X        end;
X    4 : begin
X         writeln ('Expecting a "=" operator');
X         writeln ('Got instead ',token)
X        end;
X    5 : begin
X         writeln ('Expecting either a ":" or "=" operator');
X         writeln ('Instead it got ',token)
X        end;
X    6 : begin
X         writeln ('Expecting a boolean equation');
X         writeln ('Unexpected token ',token)
X        end;
X    7 : begin
X         writeln ('Exceeded total or-terms');
X         writeln ('Output =',outtable[nexout].outname);
X        end;
X    8 : begin
X         writeln (token,' not a valid input or feedback factor')
X        end;
X    9 : begin
X         writeln ('Expecting ";" at end of equation')
X        end;
X   10 : begin
X         writeln ('This device is not capable of this function')
X        end;
X   11 : begin
X         writeln ('This device is not capable of true-form output ',token)
X        end;
X   12 : begin
X         writeln ('Not a valid output pin for ',token);
X        end;
X    otherwise:
X        writeln ('software error in error routine')
X   end
X  end; {error}
X
X procedure andterm;
X  begin
```

```
X   gettoken;
X   case sym of
X    ident :
X     begin
X      search (3);     {find pin number attached to this signal name}
X      if not found
X       then begin error(1); gettoken end
X       else begin
X             j := fusetoinp[palkind].transfer[pointer]; {get fuse number}
X             if j < 0
X              then error (8)   {not a valid input or feedback factor}
X              else outtable[outindex].matrix[iterm,j]:= '1';
X             gettoken;
X             if sym = andoperator then andterm  {call back recursively}
X            end
X     end;
X    invert :
X     begin
X      gettoken;     {get signal name}
X      if sym = ident
X       then
X        begin
X         search (3);     {find pin number attached to this signal name}
X         if not found
X          then begin error(1); gettoken end
X          else begin
X                j := fusetoinp[palkind].transfer[pointer]; {get fuse number}
X                if j < 0
X                 then error (8)   {not a valid input or feedback factor}
X                 else begin
X                       j := j + 1;    {increment fuse number}
X                       outtable[outindex].matrix[iterm,j]:= '1'
X                      end;
X                gettoken;
X                if sym = andoperator then andterm  {call back recursively}
X               end
X        end
X       else error (3)   {expecting an identifier, i.e. signal name}
X     end;
X    otherwise:  error (2)
X   end
X  end; {andterm}
X
X procedure nodeterm (var pterm : ptermtyp);
X  begin
X   gettoken;
X   case sym of
X    ident :
X     begin
X      search (3);     {find pin number attached to this signal name}
X      if not found
X       then begin error(1); gettoken end
X       else begin
```

```
X             j := fusetoinp[palkind].transfer[pointer]; {get fuse number}
X             if j < 0
X              then error (8)   {not a valid input or feedback factor}
X              else pterm[j]:= '1';
X             gettoken;
X             if sym = andoperator then nodeterm(pterm)  {call back recursively}
X            end
X     end;
X    invert :
X     begin
X      gettoken;     {get signal name}
X      if sym = ident
X       then
X        begin
X         search (3);     {find pin number attached to this signal name}
X         if not found
X          then begin error(1); gettoken end
X          else begin
X                j := fusetoinp[palkind].transfer[pointer]; {get fuse number}
X                if j < 0
X                 then error (8)   {not a valid input or feedback factor}
X                 else begin
X                       j := j + 1;    {increment fuse number}
X                       pterm[j]:= '1'
X                      end;
X                gettoken;
X                if sym = andoperator then nodeterm(pterm)
X               end
X        end
X       else error (3)   {expecting an identifier, i.e. signal name}
X     end;
X    otherwise:  error (2)
X   end
X  end; {nodeterm}
X
X  procedure setiterm;
X   begin
X    case palkind of
X     p22vx, p16l8,
X     p20l10,p20l8 :  iterm := 2;   { all outputs have OE term }
X     p16rx:
X      case pointer of
X        19,12 : if (ptype = '5')or(ptype='6')
X                  then iterm := 2  else iterm := 1;
X        18,13 : if (ptype = '6') then iterm :=2 else iterm := 1;
X        otherwise:  iterm := 1
X      end;
X     p20rx:
X      case pointer of
X        22,15 : if (ptype = 'B')or(ptype='C')
X                  then iterm := 2  else iterm := 1;
X        21,16 : if (ptype = 'C') then iterm :=2 else iterm := 1;
X        otherwise:  iterm := 1
```

```
X      end;
X     otherwise: iterm := 1
X    end
X   end; {setiterm}
X
X  procedure getterms;
X   begin
X    case palkind of
X     p10l8,p12l10:
X        totalterms := 2;
X     p14l4,p20l10:
X        totalterms := 4;
X     p12l6:
X        if (pointer = 18) or (pointer = 13)
X          then totalterms := 4
X          else totalterms := 2;
X     p14l8:
X        if (pointer = 22) or (pointer = 15)
X          then totalterms := 4
X          else totalterms := 2;
X     p16l6:
X        if (pointer = 19) or (pointer = 18)
X          then totalterms := 2
X          else totalterms := 4;
X     p18l4:
X        if (pointer = 19) or (pointer = 18)
X          then totalterms := 4
X          else totalterms := 6;
X     p22vx:
X        case pointer of
X         23,14 :  totalterms := 9;
X         22,15 :  totalterms := 11;
X         21,16 :  totalterms := 13;
X         20,17 :  totalterms := 15;
X         19,18 :  totalterms := 17;
X         otherwise:  writeln ('Software error in procedure getterms!')
X        end;
X     otherwise:
X        totalterms := 8
X    end
X   end; {getterms}
X
X procedure map (typ : char);
X  var i,j : integer;
X  begin {map}
X   case typ of
X    '0' : {initialize a new output}
X        begin
X         {first find out if output already has been defined, that is if
X           output has an enable previously defined }
X         found := false;
Xwriteln ('output : ',token,' nexout=',nexout);
X         for i:=1 to nexout do
```

```
X           with outtable[i] do
X            if outname = token
X             then begin
X                   found := true;     outindex := i
X                  end;
X         getterms;   {find out how many or-terms this output has }
X         setiterm;   {find out where to start orterms }
X         if not found
X          then begin
X                nexout := nexout + 1;
X                outtable[nexout].outnumb := pointer; {store output pin number}
X                outtable[nexout].outname := token;   {store output name }
X                for i:=1 to maxterms do
X                 for j:=1 to maxcols do
X                  outtable[nexout].matrix[i,j]:= '0';
X                outtable[nexout].outkind := nonreg;     {default}
X                if iterm > 1 then outtable[nexout].matrix[1,1]:= 'H';
X                case ptype of
X                 '2' :  outtable[nexout].form := high;
X                 otherwise:  outtable[nexout].form := low
X                end;
X                outindex := nexout
X               end
X        end
X   end
X  end; {map}
X
X procedure orterms;
X  begin
X   andterm;
X   if sym = oroperator
X    then begin
X          iterm := iterm + 1;
X          if iterm > totalterms
X           then error (7)
X           else orterms
X         end
X    else begin    {mark termination of equation}
X          iterm := iterm + 1;
X          outtable[outindex].matrix[iterm,1]:= 'X'
X         end
X  end; {orterms}
X
X procedure getmatrix;
X  begin {getmatrix}
X   case sym of
X     enable :
X      begin
X       gettoken;
X       if sym = ident
X        then begin
X              search (3);  {find out pin number}
X              if not found
```

```
X               then error (1)
X               else begin
X                      map ('0');   {create an output description database}
X                      if iterm > 1 then
X                       begin
X                        outtable[nexout].matrix[1,1]:= '0'; {clear possible H}
X                        gettoken;    {get equal sign}
X                        if sym = eql
X                         then begin
X                                iterm := 1;
X                                andterm;
X                                if sym = semicolon
X                                 then begin
X                                       gettoken;   {find out next step}
X                                       if sym <> ends then getmatrix
X                                      end
X                                 else error (9)   {missing semicolon}
X                               end
X                          else error (4)
X                         end
X                         else error (10)  { Output has no OE term }
X                    end
X             end
X       else error (3)          {expecting a signal name}
X     end;
X    clear:
X     begin
X      {find out if this is a Pal 22v10}
X      if ptype <> 'D'
X       then error (10)
X       else begin
X             gettoken;  { read dummy pseudo pin name }
X             gettoken;  { get equal sign }
X             if (sym = eql)
X              then begin
X                     ar[1]:= '0';   {erase default}
X                     nodeterm (ar);
X                     if sym = semicolon
X                      then begin
X                            gettoken;  {find out next step}
X                            if sym <> ends then getmatrix
X                           end
X                      else error (9)
X                    end
X              else error (4)
X             end
X     end;
X    preset:
X     begin
X      {find out if this is a Pal 22v10}
X      if ptype <> 'D'
X       then error (10)
X       else begin
```

```
X              gettoken;  { read dummy pseudo pin name }
X              gettoken;  { get equal sign }
X              if (sym = eql)
X               then begin
X                     sp[1]:= '0';   {erase default}
X                     nodeterm (sp);
X                     if sym = semicolon
X                      then begin
X                            gettoken;  {find out next step}
X                            if sym <> ends then getmatrix
X                           end
X                      else error (9)
X                    end
X               else error (4)
X             end
X      end;
X     ident : {a min-term equation}
X      begin
X       if (ptype = 'D') or (ptype = '2')
X        then begin
X              search (3);
X              if not found
X               then error (1)
X               else begin
X                     map ('0');    {initialize new entry in the output table}
X                     outtable[nexout].form := high;      {set output pol }
X                     gettoken;  { get equal sign }
X                     case sym of
X                      colon : {it is a registered output }
X                       begin
X                        gettoken;     {get equal sign}
X                        if (sym = eql)
X                         then
X                          begin
X                           outtable[outindex].outkind := reg;
X                           orterms;
X                           if sym = semicolon
X                            then begin
X                                  gettoken; {find out next step}
X                                  if sym <> ends then getmatrix
X                                 end
X                             else error (9)
X                           end
X                         else error (4)
X                       end;
X                      eql : {it is a non_registered output }
X                       begin
X                        outtable[outindex].outkind := nonreg;
X                        orterms;
X                        if sym = semicolon
X                         then begin
X                               gettoken; {find out next step}
X                               if sym <> ends then getmatrix
```

```
X                               end
X                             else error (9)
X                            end;
X                           otherwise:  error (5)
X                          end
X                        end
X              end
X        else error (11)    {this device is not capable of true form output}
X      end;
X     invert: {a max-term equation}
X      begin
X       gettoken;    {get signal name}
X       if sym = ident
X        then begin
X              search (3);       {obtain pin number from table}
X              if (not found)
X               then error (1)
X               else begin
X                     map ('0');    {initialize new entry in the output table}
X                     gettoken;  { get equal sign }
X                     case sym of
X                      colon : {it is a registered output }
X                       begin
X                        gettoken;     {get equal sign}
X                        if (sym = eql)
X                         then
X                          begin
X                           outtable[outindex].outkind := reg;
X                           orterms;
X                           if sym = semicolon
X                            then begin
X                                  gettoken; {find out next step}
X                                  if sym <> ends then getmatrix
X                                 end
X                             else error (9)
X                           end
X                          else error (4)
X                       end;
X                      eql : {it is a non_registered output }
X                       begin
X                        outtable[outindex].outkind := nonreg;
X                        orterms;
X                        if sym = semicolon
X                         then begin
SHAR_EOF
echo "End of part 1, continue with part 2"
echo "2" > s2_seq_.tmp
exit 0


From 75138394076730210
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,1e3e0ab64ddcef2a,start
```

X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-01-19 01:14:14 PST
Path: utzoo!attcan!uunet!lll-winken!ncis.llnl.gov!ncis!helios.ee.lbl.gov!pasteur!
ucbvax!agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i095: passm -- Logic circuit assembler part02/02
Keywords: shar, Pascal
Message-ID: <6030@saturn.ucsc.edu>
Date: 19 Jan 89 09:14:14 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 529
Approved: koreth@ssyx.ucsc.edu

Submitted-by: sun.com!skywest!BRENES (Erasmo Brenes)
Posting-number: Volume 1, Issue 95
Archive-name: passm/part02

```
#!/bin/sh
# this is part 2 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file PASSM.PAS continued
#
CurArch=2
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> PASSM.PAS
X                                gettoken; {find out next step}
X                                if sym <> ends then getmatrix
X                               end
X                          else error (9)
X                        end;
X                       otherwise:  error (5)
X                      end
X                    end
X             end
X        else error (3)
X      end;
X     otherwise: error (6) {fatal error, not a valid equation}
X   end {case of sym}
X  end; {getmatrix}
X
X procedure convrt (var numbr1 : Long_Integer; var ihex : string2);
X  var
X   i : integer;
X   res,zero,a : Long_Integer;
```

```
X   vel : Long_Integer;
X  begin
X   zero := ord ('0');
X   a := ord ('A');
X   i := 0;
X   ihex [1]:= '0';     ihex [2]:= '0';
X   ihex [3]:= '0';     ihex [4]:= '0';
X   vel := numbr1 & $0000ffff;
X   repeat
X    res := vel mod 16;
X    vel := vel div 16;
X    if res < 10
X      then  ihex [4-i]:= chr(res + zero)
X      else  ihex [4-i]:= chr(res + a - 10);
X    i:= i + 1
X   until (vel = 0)
X  end; {convrt}
X
X procedure dojedec;
X{This procedure generates the jedec file based on information from getmatrix}
X  var
X   stx,etx : char;
X   i,j,k : integer;
X   totalcol,totalfuse,nouts,firstp : integer;
X   outn,bitn, n : integer;
X   checksum : Long_Integer;
X   power2 : array [1..8] of integer;
X   scksum : string2;
X   finish : boolean;
X  begin
X   i:= 2;       stx:= chr(i);   i:= 3;  etx := chr(i);
X   power2[1]:= 1;
X   for i:=2 to 8 do power2[i]:= 2*power2[i-1];
X   pal16 := false;
X   case palkind of
X    p10l8: begin
X            pal16 := true;      totalcol := 20;
X            totalfuse := 320;   nouts := 8;     firstp := 19;
X           end;
X    p12l6: begin
X            pal16 := true;      totalcol := 24;
X            totalfuse := 384;   nouts := 6;     firstp := 18;
X           end;
X    p14l4: begin
X            pal16 := true;      totalcol := 28;
X            totalfuse := 448;   nouts := 4;     firstp := 17;
X           end;
X    p16l2: begin
X            pal16 := true;      totalcol := 32;
X            totalfuse := 512;   nouts := 2;     firstp := 16;
X           end;
X    p16l8,p16rx:
X           begin
```

```
X            pal16 := true;      totalcol := 32;
X            totalfuse := 2048;  nouts := 8;     firstp := 19;
X           end;
X    p12l10:begin
X            totalcol := 24;     totalfuse := 480;
X            nouts := 10;        firstp := 23;
X           end;
X    p14l8: begin
X            totalcol := 28;     totalfuse := 560;
X            nouts := 8;         firstp := 22;
X           end;
X    p16l6: begin
X            totalcol := 32;     totalfuse := 640;
X            nouts := 6;         firstp := 21;
X           end;
X    p18l4: begin
X            totalcol := 36;     totalfuse := 720;
X            nouts := 4;         firstp := 20;
X           end;
X    p20l2: begin
X            totalcol := 40;     totalfuse := 640;
X            nouts := 2;         firstp := 19;
X           end;
X    p20l10:begin
X            totalcol := 40;     totalfuse := 1600;
X            nouts := 10;        firstp := 23;
X           end;
X    p20l8,p20rx:
X           begin
X            totalcol := 40;     totalfuse := 2560;
X            nouts := 8;         firstp := 22;
X           end;
X    p22vx: begin
X            totalcol := 44;     totalfuse := 5828;
X            nouts := 10;        firstp := 23;
X           end
X   end; {case of ptype}
X   write (source,stx);     {write start of text}
X   write (source,'Portable Pal Assembler Jedec Output for device :');
X   writeln (source,pdevice,'*');
X   if pal16 then  write (source,'QP20* ')
X            else  write (source,'QP24* ');
X   writeln (source,'QF',totalfuse:4,'*');
X   write (source,'L0000');
X   {at this point in time, it is assumed that every output signal has a valid
X    output pin }
X   checksum := 0;    bitn:= 0;    {initialize checksum variables}
X   if palkind = p22vx
X    then {let us take care of special nodes}
X     begin
X      writeln(source);
X      if ar[1] = 'L'
X       then begin
```

```
X              for k:=1 to totalcol do
X               write (source,'0');    {unblown fuse}
X              bitn := bitn + totalcol   {increment fuse count}
X             end
X       else begin
X             for k:=1 to totalcol do
X              if ar[k] = '1' then begin
X                                   write (source,'0');
X                                   bitn := bitn + 1
X                                  end
X                             else begin
X                                   write (source,'1');
X                                   n := (bitn mod 8) + 1;
X                                   checksum := checksum + power2[n];
X                                   bitn := bitn + 1
X                                  end
X             end
X     end;
X   for i:= 1 to nouts do
X    begin
X    {first find out if there is an output with such pin}
X     outn := 0;    {default to no output defined for current pin}
X     pointer := firstp;
X     getterms;   {find out how many or-terms for this output}
X     for j:=1 to nexout do
X      with outtable[j] do
X       if outnumb = firstp   then outn := j;
X     if outn = 0
X      then begin  {no output defined for this output pin}
X            for j:=1 to totalterms do
X             begin
X              writeln (source);
X              for k:=1 to totalcol do
X               write (source,'0');    {unblown fuse}
X              bitn := bitn + totalcol   {increment fuse count}
X             end
X           end
X      else begin  {there is an output definition for this output pin}
X            finish := false;
X            for j:=1 to totalterms do
X             begin
X              writeln (source);     {terminate previous line}
X              with outtable[outn] do
X               if (matrix[j,1] <> 'X') and not finish
X                then
X                 for k:=1 to totalcol do
X                  if matrix[j,k] = '1' then begin
X                                             write (source,'0');
X                                             bitn := bitn + 1
X                                            end
X                                       else begin
X                                             write (source,'1');
X                                             n := (bitn mod 8) + 1;
```

```
X                                              checksum := checksum + power2[n];
X                                              bitn := bitn + 1
X                                             end
X               else begin
X                     for k:=1 to totalcol do write (source,'0');
X                     bitn := bitn + totalcol;
X                     finish := true     {note that this method is redundant}
X                    end
X            end
X          end;
X     firstp := firstp - 1         {step to next valid output}
X    end;
X   if palkind = p22vx
X    then {let us take care of special nodes}
X     begin
X      writeln(source);
X      if sp[1] = 'L'
X       then begin
X              for k:=1 to totalcol do
X               write (source,'0');    {unblown fuse}
X              bitn := bitn + totalcol   {increment fuse count}
X            end
X       else begin
X             for k:=1 to totalcol do
X              if sp[k] = '1' then begin
X                                   write (source,'0');
X                                   bitn := bitn + 1
X                                  end
X                             else begin
X                                   write (source,'1');
X                                   n := (bitn mod 8) + 1;
X                                   checksum := checksum + power2[n];
X                                   bitn := bitn + 1
X                                  end
X            end;
X      writeln (source,'*');  {terminate main fuse body}
X      {now let's take care of output macro cells}
X      write (source,'L5808 '); {it must be 5808 }
X      firstp := 23;
X      for i:=1 to nouts do
X       begin
X        outn := 0;
X        for j:=1 to nexout do
X         with outtable[j] do
X          if outnumb = firstp then outn := j;
X        if outn <> 0
X         then begin
X               if outtable[outn].form = high
X                then begin
X                      write (source,'1');
X                      n := (bitn mod 8) + 1;
X                      checksum := checksum + power2[n];
X                      bitn := bitn + 1
```

```
X                    end
X                else begin write(source,'0'); bitn := bitn + 1 end;
X               if outtable[outn].outkind = reg
X                then begin write(source,'0'); bitn := bitn + 1 end
X                else begin
X                      write (source,'1');
X                      n := (bitn mod 8) + 1;
X                      checksum := checksum + power2[n];
X                      bitn := bitn + 1
X                     end
X              end
X         else begin
X               write (source,'00');     bitn := bitn + 2
X              end;
X        firstp := firstp - 1   {get next valid output}
X       end;
X      writeln (source,'*')
X     end
X    else writeln (source,'*');  {terminate fuse list}
X   convrt (checksum,scksum);
X   writeln (source,'C',scksum,'*');
X   writeln (source,etx,'0000');     {write end of transmission}
X  end; {dojedec}
X
X begin { plassm }
X  nexout := 0;
X  reserved[1]:= 'device         ';     reserved[2]:= 'pin           ';
X  reserved[3]:= 'equations      ';     reserved[4]:= 'module        ';
X  reserved[5]:= 'flag           ';     reserved[6]:= 'title         ';
X  reserved[7]:= 'node           ';     reserved[8]:= 'istype        ';
X  reserved[9]:= 'macro          ';     reserved[10]:='ENABLE        ';
X  reserved[11]:='RESET          ';     reserved[12]:='PRESET        ';
X  reserved[13]:='end            ';
X  wsym [1]:= device;    wsym[2]:= pin;  wsym[3]:= equations;
X  wsym [4]:= module;    wsym[5]:= flag; wsym[6]:= title;
X  wsym [7]:= node;      wsym[8]:= stype; wsym[9]:= macro;
X  wsym [10]:= enable;   wsym[11]:= clear; wsym[12]:= preset;
X  wsym [13]:= ends;
X  palknds[1]:= '1';     pals[1]:= 'p10l8         ';
X  paltyp [1]:= p10l8;
X  palknds[2]:= '1';     pals[2]:= 'p12l6         ';
X  paltyp [2]:= p12l6;
X  palknds[3]:= '1';     pals[3]:= 'p14l4         ';
X  paltyp [3]:= p14l4;
X  palknds[4]:= '1';     pals[4]:= 'p16l2         ';
X  paltyp [4]:= p16l2;
X  palknds[5]:= '2';     pals[5]:= 'p10h8         ';
X  paltyp [5]:= p10l8;
X  palknds[6]:= '2';     pals[6]:= 'p12h6         ';
X  paltyp [6]:= p12l6;
X  palknds[7]:= '2';     pals[7]:= 'p14h4         ';
X  paltyp [7]:= p14l4;
X  palknds[8]:= '2';     pals[8]:= 'p16h2         ';
```

```
X  paltyp [8]:= p16l2;
X  palknds[9]:= '3';     pals[9]:= 'p16l8          ';
X  paltyp [9]:= p16l8;
X  palknds[10]:= '4';    pals[10]:= 'p16r8          ';
X  paltyp [10]:= p16rx;
X  palknds[11]:= '5';    pals[11]:= 'p16r6          ';
X  paltyp [11]:= p16rx;
X  palknds[12]:= '6';    pals[12]:= 'p16r4          ';
X  paltyp [12]:= p16rx;
X  palknds[13]:= '7';    pals[13]:= 'p12l10         ';
X  paltyp [13]:= p12l10;
X  palknds[14]:= '7';    pals[14]:= 'p14l8          ';
X  paltyp [14]:= p14l8;
X  palknds[15]:= '7';    pals[15]:= 'p16l6          ';
X  paltyp [15]:= p16l6;
X  palknds[16]:= '7';    pals[16]:= 'p18l4          ';
X  paltyp [16]:= p18l4;
X  palknds[17]:= '7';    pals[17]:= 'p20l2          ';
X  paltyp [17]:= p20l2;
X  palknds[18]:= '8';    pals[18]:= 'p20l10         ';
X  paltyp [18]:= p20l10;
X  palknds[19]:= '9';    pals[19]:= 'p20l8          ';
X  paltyp [19]:= p20l8;
X  palknds[20]:= 'A';    pals[20]:= 'p20r8          ';
X  paltyp [20]:= p20rx;
X  palknds[21]:= 'B';    pals[21]:= 'p20r6          ';
X  paltyp [21]:= p20rx;
X  palknds[22]:= 'C';    pals[22]:= 'p20r4          ';
X  paltyp [22]:= p20rx;
X  palknds[23]:= 'D';    pals[23]:= 'p22v10         ';
X  paltyp [23]:= p22vx;
X  { pin number to fuse column transform }
X  with fusetoinp [p10l8] do
X   begin
X    for i:=1 to maxpins do transfer[i]:= -1;
X    transfer[1]:= 3;    transfer[2]:= 1;        transfer[3]:= 5;
X    transfer[4]:= 7;    transfer[5]:= 9;        transfer[6]:= 11;
X    transfer[7]:= 13;   transfer[8]:= 15;       transfer[9]:= 17;
X    transfer[11]:= 19
X   end;
X  with fusetoinp [p12l6] do
X   begin
X    for i:=1 to maxpins do transfer[i]:= -1;
X    transfer[1]:= 3;    transfer[2]:= 1;        transfer[3]:= 5;
X    transfer[4]:= 9;    transfer[5]:= 11;       transfer[6]:= 13;
X    transfer[7]:= 15;   transfer[8]:= 17;       transfer[9]:= 21;
X    transfer[11]:= 23;  transfer[12]:= 19;      transfer[19]:= 7
X   end;
X  with fusetoinp [p14l4] do
X   begin
X    for i:=1 to maxpins do transfer[i]:= -1;
X    transfer[1]:= 3;    transfer[2]:= 1;        transfer[3]:= 5;
X    transfer[4]:= 9;    transfer[5]:= 13;       transfer[6]:= 15;
```

```
X    transfer[7]:= 17;   transfer[8]:= 21;       transfer[9]:= 25;
X    transfer[11]:= 27;  transfer[12]:= 23;      transfer[13]:= 19;
X    transfer[18]:= 11;  transfer[19]:= 7
X   end;
X  with fusetoinp [p16l2] do
X   begin
X    for i:=1 to maxpins do transfer[i]:= -1;
X    transfer[1]:= 3;    transfer[2]:= 1;        transfer[3]:= 5;
X    transfer[4]:= 9;    transfer[5]:= 13;       transfer[6]:= 17;
X    transfer[7]:= 21;   transfer[8]:= 25;       transfer[9]:= 29;
X    transfer[11]:= 31;  transfer[12]:= 27;      transfer[13]:= 23;
X    transfer[14]:= 19;  transfer[17]:= 15;      transfer[18]:= 11;
X    transfer[19]:= 7
X   end;
X  with fusetoinp [p16l8] do
X   begin
X    for i:=1 to maxpins do transfer[i]:= -1;
X    transfer[1]:= 3;    transfer[2]:= 1;        transfer[3]:= 5;
X    transfer[4]:= 9;    transfer[5]:= 13;       transfer[6]:= 17;
X    transfer[7]:= 21;   transfer[8]:= 25;       transfer[9]:= 29;
X    transfer[11]:= 31;  transfer[13]:= 27;      transfer[14]:= 23;
X    transfer[15]:= 19;  transfer[16]:= 15;      transfer[17]:= 11;
X    transfer[18]:= 7
X   end;
X  with fusetoinp [p16rx] do
X   begin
X    for i:=1 to maxpins do transfer[i]:= -1;
X                          transfer[2]:= 1;       transfer[3]:= 5;
X    transfer[4]:= 9;      transfer[5]:= 13;      transfer[6]:= 17;
X    transfer[7]:= 21;     transfer[8]:= 25;      transfer[9]:= 29;
X                          transfer[12]:= 31;     transfer[13]:= 27;
X    transfer[14]:= 23;    transfer[15]:= 19;     transfer[16]:= 15;
X    transfer[17]:= 11;    transfer[18]:= 7;      transfer[19]:= 3
X   end;
X  with fusetoinp [p12l10] do
X   begin
X    for i:=1 to maxpins do transfer[i]:= -1;
X    transfer[1]:= 3;    transfer[2]:= 1;        transfer[3]:= 5;
X    transfer[4]:= 7;    transfer[5]:= 9;        transfer[6]:= 11;
X    transfer[7]:= 13;   transfer[8]:= 15;       transfer[9]:= 17;
X    transfer[10]:= 19;  transfer[11]:= 21;      transfer[13]:= 23
X   end;
X  with fusetoinp [p14l8] do
X   begin
X    for i:=1 to maxpins do transfer[i]:= -1;
X    transfer[1]:= 3;    transfer[2]:= 1;        transfer[3]:= 5;
X    transfer[4]:= 9;    transfer[5]:= 11;       transfer[6]:= 13;
X    transfer[7]:= 15;   transfer[8]:= 17;       transfer[9]:= 19;
X    transfer[10]:= 21;  transfer[11]:= 25;
X    transfer[13]:= 27;  transfer[14]:= 23;      transfer[23]:= 7
X   end;
X  with fusetoinp [p16l6] do
X   begin
```

```
X    for i:=1 to maxpins do transfer[i]:= -1;
X    transfer[1]:= 3;    transfer[2]:= 1;        transfer[3]:= 5;
X    transfer[4]:= 9;    transfer[5]:= 13;       transfer[6]:= 15;
X    transfer[7]:= 17;   transfer[8]:= 19;       transfer[9]:= 21;
X    transfer[10]:= 25;  transfer[11]:= 29;
X    transfer[13]:= 31;  transfer[14]:= 27;      transfer[15]:= 23;
X    transfer[22]:= 11;  transfer[23]:= 7
X   end;
X  with fusetoinp [p18l4] do
X   begin
X    for i:=1 to maxpins do transfer[i]:= -1;
X    transfer[1]:= 3;    transfer[2]:= 1;        transfer[3]:= 5;
X    transfer[4]:= 9;    transfer[5]:= 13;       transfer[6]:= 17;
X    transfer[7]:= 19;   transfer[8]:= 21;       transfer[9]:= 25;
X    transfer[10]:= 29;  transfer[11]:= 33;
X    transfer[13]:= 35;  transfer[14]:= 31;     transfer[15]:= 27;
X    transfer[16]:= 23;  transfer[21]:= 15;     transfer[22]:= 11;
X    transfer[23]:= 7
X   end;
X  with fusetoinp [p20l2] do
X   begin
X    for i:=1 to maxpins do transfer[i]:= -1;
X    transfer[1]:= 3;    transfer[2]:= 1;        transfer[3]:= 5;
X    transfer[4]:= 9;    transfer[5]:= 13;       transfer[6]:= 17;
X    transfer[7]:= 21;   transfer[8]:= 25;       transfer[9]:= 29;
X    transfer[10]:= 33;  transfer[11]:= 37;
X    transfer[13]:= 39;  transfer[14]:= 35;      transfer[15]:= 31;
X    transfer[16]:= 27;  transfer[17]:= 23;
X    transfer[20]:= 19;  transfer[21]:= 15;      transfer[22]:= 11;
X    transfer[23]:= 7
X   end;
X  with fusetoinp [p20l10] do
X   begin
X    for i:=1 to maxpins do transfer[i]:= -1;
X    transfer[1]:= 3;    transfer[2]:= 1;        transfer[3]:= 5;
X    transfer[4]:= 9;    transfer[5]:= 13;       transfer[6]:= 17;
X    transfer[7]:= 21;   transfer[8]:= 25;       transfer[9]:= 29;
X    transfer[10]:= 33;  transfer[11]:= 37;
X    transfer[13]:= 39;  transfer[15]:= 35;      transfer[16]:= 31;
X    transfer[17]:= 27;  transfer[18]:= 23;      transfer[19]:= 19;
X    transfer[20]:= 15;  transfer[21]:= 11;      transfer[22]:= 7
X   end;
X  with fusetoinp [p20l8] do
X   begin
X    for i:=1 to maxpins do transfer[i]:= -1;
X    transfer[1]:= 3;    transfer[2]:= 1;        transfer[3]:= 5;
X    transfer[4]:= 9;    transfer[5]:= 13;       transfer[6]:= 17;
X    transfer[7]:= 21;   transfer[8]:= 25;       transfer[9]:= 29;
X    transfer[10]:= 33;  transfer[11]:= 37;
X    transfer[13]:= 39;  transfer[14]:= 35;
X    transfer[16]:= 31;  transfer[17]:= 27;      transfer[18]:= 23;
X    transfer[19]:= 19;  transfer[20]:= 15;      transfer[21]:= 11;
X    transfer[23]:= 7
```

```
X   end;
X  with fusetoinp [p20rx] do
X   begin
X    for i:=1 to maxpins do transfer[i]:= -1;
X                        transfer[2]:= 1;        transfer[3]:= 5;
X    transfer[4]:= 9;    transfer[5]:= 13;       transfer[6]:= 17;
X    transfer[7]:= 21;   transfer[8]:= 25;       transfer[9]:= 29;
X    transfer[10]:= 33;  transfer[11]:= 37;
X                        transfer[14]:= 39;      transfer[15]:= 35;
X    transfer[16]:= 31;  transfer[17]:= 27;      transfer[18]:= 23;
X    transfer[19]:= 19;  transfer[20]:= 15;      transfer[21]:= 11;
X    transfer[22]:= 7;   transfer[23]:= 3
X   end;
X  with fusetoinp [p22vx] do
X   begin
X    for i:=1 to maxpins do transfer[i]:= -1;
X    transfer[1]:= 1;    transfer[2]:= 5;        transfer[3]:= 9;
X    transfer[4]:= 13;   transfer[5]:= 17;       transfer[6]:= 21;
X    transfer[7]:= 25;   transfer[8]:= 29;       transfer[9]:= 33;
X    transfer[10]:= 37;  transfer[11]:= 41;
X    transfer[13]:= 43;  transfer[14]:= 39;      transfer[15]:= 35;
X    transfer[16]:= 31;  transfer[17]:= 27;      transfer[18]:= 23;
X    transfer[19]:= 19;  transfer[20]:= 15;      transfer[21]:= 11;
X    transfer[22]:= 7;   transfer[23]:= 3
X   end;
X  tab := chr(9);        nexin := 0;     Abort := false;    ch:= blank;
X  writeln;
X  writeln ('  Portable Pal Assembler');
X  writeln ('  Rev.1  Sep 1988');
X  writeln ('  By: Erasmo Brenes ');
X  writeln ('  (c) Copyright 1987,1988');
X  writeln;
X  for i:=1 to 80 do filspc[i]:= blank;
X  for i:=1 to maxcols do begin ar[i]:= '0'; sp[i]:= '0'  end;
X  { Default to inactive for ar and sp}
X  ar[1]:= 'L';  sp[1]:= 'L';
X  write ('Enter source filename_');
X  readln (filspc);
X  reset(source,filspc);
X  getnames;
X{*** diag print ***}
X for i:= 1 to nexin do
X  with symtable[i] do
X   writeln ('pin name= ',name,' pin#=',pinn:3);
X i:= 1;
X if not Abort
X  then begin
X        gettoken;    {get first token before calling getmatrix}
X        getmatrix;
X        close (source);   {release previous handle}
X        while (filspc[i] <> '.') do i:= i + 1;
X        i:= i + 1;    j:= i;
X        filspc[i]:= 'j';      i:= i + 1;
```

```
X        filspc[i]:= 'e';      i:= i + 1;
X        filspc[i]:= 'd';
X        rewrite (source,filspc);
X        if not Abort then dojedec;
X        writeln ('Press any key to return');
X        while (not Keypress) do begin end    {ie do nothing}
X       end
X end.
SHAR_EOF
chmod 0600 PASSM.PAS || echo "restore of PASSM.PAS fails"
rm -f s2_seq_.tmp
echo "You have unpacked the last part"
exit 0


From 5719483032190531318
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,3933de084489c99b,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-01-20 10:05:51 PST
Path: utzoo!attcan!uunet!lll-winken!lll-lcc!ncis.llnl.gov!helios.ee.lbl.gov!
pasteur!agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i096: leolife -- Game of Life
Keywords: shar, C
Message-ID: <6044@saturn.ucsc.edu>
Date: 20 Jan 89 18:05:51 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 440
Approved: koreth@ssyx.ucsc.edu

Submitted-by: leo@philmds.dts.philips.nl (Leo de Wit)
Posting-number: Volume 1, Issue 96
Archive-name: leolife

[The binaries have been posted to the binaries group. -sg]

: This is a shar archive.  Extract with sh, not csh.
: This archive ends with exit, so do not worry about trailing junk.
echo 'Extracting life.c'
sed 's/^X//' > life.c << '+ END-OF-FILE life.c'
X/*****************************************************************************
X *                                                                           *
X *                     COPYRIGHT (C) 1988    L.J.M. de Wit                   *
X *                     ALL RIGHTS RESERVED                                   *
X *                                                                           *
X *                     life.c   version 1.0 of 11 Nov 1988                   *
X *                                                                           *
X * This software may be used and distributed freely if not used commercially  *
X * and the originator (me) is mentioned.                                     *
X *                                                                           *
X *****************************************************************************
```

```
X *
X * NAME
X *    life   - the game of life (graphics demo if you want)
X *
X * SYNTAX
X *    life.ttp [-d|-D] [-w wait] [01-pattern ...]
X *
X * DESCRIPTION
X *    This program barely needs comments. 'Generations' are created on a grid,
X *    on a step by step basis. The criterium used for a cell to survive
X *    (resp. die), is that it has between BOT (3) and TOP (4) neighbours
X *    (resp. has not).
X *
X *    The program runs only in high and medium resolution modes (so everyone
X *    should be able to run it); not supporting low res is entirely due to
X *    laziness on my part 8-), but since no colours are involved (yet) it
X *    seemed an unnecessary addition.
X *
X *    For both resolutions supported you can choose a coarse grid (25 x 40)
X *    and a fine one (50 x 80); default is coarse, and the -d or -D (double)
X *    option selects the fine grid.
X *
X *    The -w or -W option lets you specify an amount of time to wait between
X *    generations (default is nowait).
X *
X *    Any remaining arguments are taken to be the initial patterns to be placed
X *    on the grid, a '0' representing an empty cell, a '1' a filled one. These
X *    patterns are placed on subsequent rows, near the middle of the screen.
X *    Some pattern sets generate repetitive populations, other die away, still
X *    other generate nice ever-changing patterns. Experiment. Symmetric sets
X *    are (possibly) favorite; this means using symmetric patterns, with the
X *    first matching the last, the second the last but one, etc., and using an
X *    odd number of patterns in the coarse grid case and an even number of
X *    patterns in the fine grid case. You could alternative use point symmetry
X *    instead of line symmetry; figure out for yourself how to arrange this.
X *    If there are no remaining arguments after any flags, a random pattern is
X *    generated.
X *
X *    To finish the game, 'press any key ...'
X *
X * EXAMPLE
X *    life -d -w 20 011110 110011 101101 001100 101101 110011 011110
X *
X *    will have a fine grid, a wait pause of 20, and an initial pattern set
X *    that looks like:
X *                                  ****
X *                                **  **
X *                                * ** *
X *                                  **
X *                                * ** *
X *                                **  **
X *                                 ****
X * DECISIONS
```

```
X *    Screen access is done directly, for speed reasons (we must be able to
X *    print several hundreds of characters, several times per second, if we
X *    want screen updates to appear - almost - immediate).
X *    On the other hand, C proved to be quite fast enough, so there was no
X *    need to resort to assembler.
X *    Each resolution and each mode (coarse/fine) demands it own treatment,
X *    so four separate (functional identical) functions were written, one
X *    for each of them.
X *
X *    By using a private startup routine (not included) instead of the
X *    compiler supplied one, the size of the program could be kept
X *    considerably smaller (not using stdio).
X */
X
X#include <osbind.h>
X
X#define FALSE 0
X#define TRUE  1
X
X#define CPUTC(c) bios(3,2,c)              /* Write char c to console */
X#define SC_24 0xffffff                    /* Mask 24 bits */
X#define BOT 3                             /* Min. # of neighbours to survive */
X#define TOP 4                             /* Max. # of neighbours to survive */
X
X#define B_P_ROW  1280                     /* Bytes per screen row */
X#define B_P_DROW  640                     /* Bytes per half screen row */
X#define W_P_ROW   640                     /* Words per screen row */
X#define W_P_DROW  320                     /* Words per half screen row */
X#define L_P_ROW   320                     /* Longs per screen row */
X#define L_P_DROW  160                     /* Longs per half screen row */
X
X#define B_P_HLIN   80                     /* Bytes per hires line */
X#define W_P_HLIN   40                     /* Words per hires line */
X#define L_P_MLIN   40                     /* Longs per meres line */
X
X#define STDROWS    25                     /* Default # of grid rows */
X#define STDCOLS    40                     /* Default # of grid columns */
X#define MAXARRSIZ (STDROWS*STDCOLS*4)     /* Max.gridsize for either gridtype */
X
X#define BROW(bp,r) (*(bp) + Cols * (r))   /* Start of row r in block *bp   */
X
X/* create some unsigned types */
Xtypedef unsigned char  uchar;
Xtypedef unsigned short ushort;
Xtypedef unsigned long  ulong;
X
Xtypedef uchar grid[MAXARRSIZ];           /* A 'grid of cells' type */
X
Xstatic grid *oldp, *newp;                 /* Pointers to old and new grid */
X
Xstatic short Rows;                        /* Actual # of rows in grid */
Xstatic short Cols;                        /* Actual # of columns in grid */
X
```

```
Xstatic ulong *start;                       /* Points to logical screen start */
X
Xint _mneed = 60000;                        /* For Lattice C: restricts the   */
X                                           /* memory used by the program, so */
X                                           /* that Mallocs will find enough  */
X                                           /* memory left.                   */
X
Xstatic ushort dsized = FALSE;              /* Boolean: fine grid used? */
X
Xstatic void init_grid(),                   /* Initiate one grid        */
X            calc_grid(),                   /* Calculate next grid      */
X            show_s1grid(),                 /* Show coarse grid med res */
X            show_d1grid(),                 /* Show fine grid med res   */
X            show_s2grid(),                 /* Show coarse grid hi res  */
X            show_d2grid();                 /* Show fine grid hi res    */
X
Xmain(argc,argv)
Xint argc;
Xchar **argv;
X{
X   short cntr = 0;                         /* # of generations                */
X   grid *swap;                            /* Temporary to change grid ptrs   */
X   register ulong *upn, *upo, uv;         /* For clearing grids initially    */
X   register short cnt;                     /* Counting clear bytes            */
X   long i, waittime = 0;                   /* Counting wait loop              */
X   void (*showfunc)();                     /* The display function being used */
X   static void (*sfuncs[])() = {          /* Possible display functions      */
X      (void (*)())0,(void (*)())0,        /* Low res not implemented         */
X      show_s1grid, show_d1grid,           /* Med res coarse and fine         */
X      show_s2grid, show_d2grid            /* Hi res coarse and fine          */
X   };
X
X   start   = (ulong *)Logbase();           /* Cur. log. screen will be used   */
X   for (--argc, argv++;
X           argc > 0 && **argv == '-';
X           --argc, argv++) {               /* Handle flag arguments           */
X      if (argv[0][1] == 'd' || argv[0][1] == 'D') {   /* Double, i.e. fine   */
X         dsized = TRUE;
X      } else if (argv[0][1] == 'w' || argv[0][1] == 'W') {  /* Wait count    */
X         if (--argc <= 0) {
X            Cconws("life: usage: life.ttp [-d|-D] [-w|-W wait] [01-pattern...]
\r\n");
X            Crawcin();
X            Pterm(1);
X         }
X         waittime = atoi(*++argv);
X         if (waittime < 0 || waittime > 10000) {
X            Cconws("life: wait must be between 0 and 10000\r\n");
X            Crawcin();
X            Pterm(1);
X         }
X      } else {                             /* Illegal flag                    */
X         Cconws("life: usage: life.ttp [-d|-D] [-w|-W wait] [01-pattern...]
```

```
\r\n");
X         Crawcin();
X         Pterm(1);
X      }
X   }
X
X   showfunc = sfuncs[2 * Getrez() + dsized]; /* Select display function      */
X   if (showfunc == (void (*)())0) {       /* Low resolution not provided for */
X      Cconws("life: must be medium/high resolution, sorry ...\r\n");
X      Crawcin();
X      Pterm(1);
X   }
X
X   Cconws("\33E\33f\33Y,1created 1988       * LIFE *       by Leo de Wit");
X   for (i = 1 << 18; --i >= 0; ) ;        /* Wait some time                  */
X   CPUTC('\33'); CPUTC('l');              /* Clear current line              */
X
X   if (dsized) {                          /* Fine grid chosen ...            */
X      Rows = STDROWS * 2;                 /* Twice as many rows ...          */
X      Cols = STDCOLS * 2;                 /* Twice as many cols ...          */
X   } else {                               /* Coarse grid chosen              */
X      Rows = STDROWS;                     /* Std. # of rows                  */
X      Cols = STDCOLS;                     /* Std. # of cols                  */
X   }
X
X   newp = (grid *)Malloc(sizeof(grid));   /* Allocate one grid               */
X   oldp = (grid *)Malloc(sizeof(grid));   /* And another                     */
X   if (newp == (grid *)-1 || oldp == (grid *)-1) {
X      Cconws("life: memory allocation failed\r\n");
X      Crawcin();
X      Pterm(1);
X   }
X   uv = 0;                                /* Now clear them both ...         */
X   upn = (ulong *)newp;
X   upo = (ulong *)oldp;
X   for (cnt = sizeof(grid)/sizeof(ulong); --cnt >= 0; ) {
X      *upn++ = *upo++ = uv;
X   }
X
X   init_grid(newp,argc,argv);             /* Initiate grid *newp             */
X   waittime <<= 10;                       /* Scale waittime reasonably       */
X
X   for (;;) {
X      (*showfunc)(newp,oldp);             /* Write to logical screen         */
X      if (((cntr++ & 0x7) == 0) && Bconstat(2)) { /* Test constat once in 8  */
X         break;
X      }
X      swap = newp; newp = oldp; oldp = swap; /* Exchange grid pointers       */
X      calc_grid(newp,oldp);               /* Calculate new grid from old     */
X      for (i = waittime; --i >= 0; ) ;    /* This is the wait loop           */
X   }
X
X   Bconin(2);                             /* Read key pressed from queue     */
```

```
X}
X
Xstatic void init_grid(bp,argc,argv)       /* Initiate grid *bp               */
Xgrid *bp;
Xint argc;
Xchar **argv;
X{
X   short len, maxlen = 0, j, i, r0, c0;
X
X   if (argc == 0) {                       /* No parameters: random pattern   */
X      short maxi = (Rows == STDROWS) ? 16 : 64; /* # of filled cells         */
X
X      for (i = 0; i < maxi; i++) {
X         short ranrow = Rows/3+((unsigned)Random())*Rows/(3*SC_24);
X         short rancol = Cols/3+((unsigned)Random())*Cols/(3*SC_24);
X
X         BROW(bp,ranrow)[rancol] = '\1';
X      }
X   } else {                                /* Pattern parameters              */
X      for (i = 0; i < argc; i++) {         /* Find longest string length      */
X         if ((len = strlen(argv[i])) > maxlen) {
X            maxlen = len;
X         }
X      }
X      r0 = (Rows - argc)/2;                /* Start row                       */
X      c0 = (Cols - maxlen)/2;              /* Start column                    */
X      for (i = 0; i < argc; i++) {         /* Handle each string              */
X         len = strlen(argv[i]);
X         for (j = 0; j < len; j++) {       /* Set each cell                   */
X            BROW(bp,r0+i)[c0+j] = argv[i][j] & 1;
X         }
X      }
X   }
X}
X
Xstatic void calc_grid(newp,oldp)          /* Calculate new grid from old     */
Xgrid *newp, *oldp;
X{
X   register short r, c;                   /* Current row, col                */
X   register uchar *cp,                    /* Ptr into new grid               */
X                  *ocp;                  /* Ptr into old grid               */
X
X   for (r = Rows; --r >= 0; ) {           /* Zero each row of grid           */
X      cp = BROW(newp,r+1);
X      for (c = Cols; --c >= 0; ) {        /* Zero each cell of row           */
X         *--cp = '\0';
X      }
X   }
X   for (r = Rows - 1; --r >= 1; ) {       /* Handle each row of old          */
X      ocp = BROW(oldp,r+1);
X      --ocp;
X      for (c = Cols - 1; --c >= 1; ) {    /* Handle each cell of old row     */
X         if (*--ocp) {                    /* If it was filled ...            */
```

```
X            cp = &BROW(newp,r-1)[c-1];    /* Incr. each neighbour's count    */
X            (*cp++)++; (*cp++)++; (*cp++)++; /* Previous row in new grid     */
X            cp += Cols - 3;
X            (*cp++)++; (*cp++)++; (*cp++)++; /* Same row in new grid         */
X            cp += Cols - 3;
X            (*cp++)++; (*cp++)++; (*cp++)++; /* Next row in new grid         */
X         }
X      }
X   }
X   for (r = Rows; --r >= 0; ) {           /* Handle each row of new grid     */
X      cp = BROW(newp,r+1);
X      for (c = Cols; --cp, --c >= 0; ) {  /* Handle each cell of row         */
X         *cp = (*cp >= BOT && *cp <= TOP);/* Set it according to accum. cnt  */
X      }
X   }
X}
X
Xstatic void show_s1grid(newp,oldp)        /* Display coarse grid med res     */
Xgrid *newp, *oldp;
X{
X   register short r, c;                   /* Current row & column            */
X   register ulong val;                   /* Value to fill with              */
X   register ulong step = L_P_MLIN;        /* Step to next screen line        */
X   register uchar *cp,                    /* Ptr into new grid               */
X                 *ocp;                   /* Ptr into old grid               */
X   register ulong *sa,                    /* Current screen address          */
X                 *basad;                 /* Screen address of end of row    */
X
X   basad = start + (Rows * L_P_ROW);
X   for (r = Rows; basad -= L_P_ROW, --r >= 0; ) {  /* Handle each row       */
X      cp = BROW(newp,r+1);
X      ocp = BROW(oldp,r+1);
X      for (c = Cols; --c >= 0; ) {        /* Handle each column              */
X         if (*--cp != *--ocp) {
X            val = -*cp;                  /* 0xffffffff : 0                  */
X            sa = basad + c;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val;
X         }
X      }
X   }
X}
X
Xstatic void show_d1grid(newp,oldp)        /* Display fine grid med res       */
Xgrid *newp, *oldp;
X{
X   register short r, c;                   /* Current row & column            */
```

```
X   register ulong val;                    /* Value to fill with             */
X   register ulong step = L_P_MLIN;        /* Step to next screen line       */
X   register ushort *sp,                   /* Ptr into new grid              */
X                  *osp;                  /* Ptr into old grid              */
X   register ulong *sa,                    /* Current screen address         */
X                  *basad;                /* Screen address of end of row   */
X
X   basad = start + (Rows * L_P_DROW);
X   for (r = Rows; basad -= L_P_DROW, --r >= 0; ) {  /* Handle each row      */
X      sp = (ushort *)BROW(newp,r+1);
X      osp = (ushort *)BROW(oldp,r+1);
X      for (c = Cols/2; --c >= 0; ) {      /* Handle each column, two a time  */
X         if (*--sp != *--osp) {
X            static ulong values[] = {0, 0xff00ff, 0xff00ff00, 0xffffffff};
X
X            val = values[(*sp & 1) + (*sp >> 7)];
X            sa = basad + c;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val;
X         }
X      }
X   }
X}
X
Xstatic void show_s2grid(newp,oldp)        /* Display coarse grid hi res     */
Xgrid *newp, *oldp;
X{
X   register short r, c;                   /* Current row & column           */
X   register ushort val;                   /* Value to fill with             */
X   register ulong step = W_P_HLIN;        /* Step to next screen line       */
X   register uchar *cp,                    /* Ptr into new grid              */
X                  *ocp;                  /* Ptr into old grid              */
X   register ushort *sa,                   /* Current screen address         */
X                   *basad;               /* Screen address of end of row   */
X
X   basad = ((ushort *)start) + (Rows * W_P_ROW);
X   for (r = Rows; basad -= W_P_ROW, --r >= 0; ) {  /* Handle each row       */
X      cp = BROW(newp,r+1);
X      ocp = BROW(oldp,r+1);
X      for (c = Cols; --c >= 0; ) {        /* Handle each column             */
X         if (*--cp != *--ocp) {
X            val = -*cp;                  /* 0xffffffff : 0                 */
X            sa = basad + c;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
```

```
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val;
X         }
X      }
X   }
X}
X
Xstatic void show_d2grid(newp,oldp)        /* Display fine grid hi res        */
Xgrid *newp, *oldp;
X{
X   register short r, c;                   /* Current row & column            */
X   register uchar val;                   /* Value to fill with              */
X   register ulong step = B_P_HLIN;        /* Step to next screen line        */
X   register uchar *cp,                   /* Ptr into new grid               */
X                 *ocp;                  /* Ptr into old grid               */
X   register uchar *sa,                   /* Current screen address          */
X                 *basad;                /* Screen address of end of row    */
X
X   basad = ((uchar *)start) + (Rows * B_P_DROW);
X   for (r = Rows; basad -= B_P_DROW, --r >= 0; ) {  /* Handle each row        */
X      cp = BROW(newp,r+1);
X      ocp = BROW(oldp,r+1);
X      for (c = Cols; --c >= 0; ) {         /* Handle each column              */
X         if (*--cp != *--ocp) {
X            val = -*cp;                   /* 0xff : 0                        */
X            sa = basad + c;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val; sa += step;
X            *sa = val;
X         }
X      }
X   }
X}
+ END-OF-FILE life.c
chmod 'u=rw,g=r,o=' 'life.c'
echo 'SENT: -rw-r-----  1 leo         19008 Jan 20 18:29 life.c'
echo -n 'RCVD: '
/bin/ls -l life.c
exit 0


From -1430669071671331589
```

X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,b79e8b1334d71958,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,b79e8b1334d71958,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-01-22 15:24:12 PST
Xref: utzoo comp.sources.atari.st:152 comp.sys.atari.st:13591
Path: utzoo!attcan!uunet!husc6!mailrus!csd4.milw.wisc.edu!bionet!agate!saturn!
ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st,comp.sys.atari.st
Subject: v02INF2: Index for comp.sources.atari.st
Message-ID: <6060@saturn.ucsc.edu>
Date: 22 Jan 89 23:24:12 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 99
Approved: koreth@ssyx.ucsc.edu (Steven Grimm)

-- Volume 1 --

```
volume1/intro           Introduction to comp.sources.atari.st
volume1/basic-uud       Uudecode in BASIC
volume1/bootcopy        Automatic file copier for RAMdisk
volume1/joyread              Joystick-reading routines
volume1/cc/part01       C compiler driver (Alcyon)
volume1/cc/part02
volume1/cc/part03
volume1/cc/part04
volume1/mkmf            Something for Absoft Fortran
volume1/make/part01          Program maintainer
volume1/make/part02
volume1/make/part03
volume1/make/part04
volume1/make/part05
volume1/make/part06
volume1/make/part07
volume1/blank           Fast disk eraser
volume1/mmdummy              Megamax dummy routines
volume1/fdum            Megamax floating-point dummy routines
volume1/idle            Screen saver
volume1/autotime        Time setter for auto folder
volume1/time            Time setter
volume1/autodisk        Copy files to RAMdisk
volume1/dvist32/part01       DVI previewer
volume1/dvist32/part02
volume1/dvist32/part03
volume1/dvist32/part04
volume1/dvist32/part05
volume1/dvist32/part06
volume1/dvist32/part07
volume1/dvist32/part08
volume1/dvist32/part09
volume1/cache           Disk cache
```

```
volume1/vdit           VDI interface with GDOS font support
volume1/flip/part01         Double-sided disk copier
volume1/flip/part02
volume1/rzsz/part01         UNIX end for zmdm, below
volume1/rzsz/part02
volume1/rzsz/part03
volume1/rzsz/part04
volume1/rzsz/part05
volume1/zmdm/part01         Zmodem file transfer program
volume1/zmdm/part02
volume1/zmdm/part03
volume1/zmdm/part04
volume1/zmdm/part05
volume1/zmdm/part06
volume1/zmdm/part07
volume1/zmdm/part08
volume1/ops5/part01         OPS5 production system in LISP
volume1/ops5/part02
volume1/ops5/part03
volume1/assist2/part01      Multifunction desk accessory
volume1/assist2/part02
volume1/errorchk       Verbose display of error codes
volume1/joinprnt       Echo console output to printer
volume1/color          Color-selection dialog box
volume1/rsctoc              Convert resource to C source
volume1/fastdump       Text-only screen dump
volume1/readme12       Online help system
volume1/newstart       Fancy printing
volume1/varlint
volume1/flip27/part01       Double-sided disk copier
volume1/flip27/part02
volume1/poplib              Shared libraries, with examples
volume1/uupc/part01         UUCP clone (pre-pre-release)
volume1/uupc/part02
volume1/uupc/part03
volume1/uupc/part04
volume1/gnufp          GNU floating point routines
volume1/scroll              Fast screen scroller
volume1/break          Stop programs with a keypress
volume1/newbreak       New version of above
volume1/timecode       Determine 68000 code speed
volume1/achlib/part01       Memory management and other routines for MadMac
volume1/achlib/part02
volume1/sttar          Unix-style "tar" utility
volume1/d110edit       D110 patch editor (MIDI)
volume1/raymovi2/part01     Raytrace movie generator
volume1/raymovi2/part02
volume1/gstlink             Faster linker for GST object files
volume1/mx2v230/part01      Multitasking kernel with utilities
volume1/mx2v230/part02
volume1/mx2v230/part03
volume1/mx2v230/part04
volume1/mx2v230/part05
```

```
volume1/mx2v230/part06
volume1/mx2v230/part07
volume1/mx2v230/part08
volume1/limiter             Limit uppermost hard disk device letter
volume1/d110edit        Patch editor for D110 synthesizer
volume1/mt32edit        Patch editor for MT32 synthesizer
volume1/starchrt/part01     Diffs to Unix starchart program for ST
volume1/starchrt/part02
volume1/passm/part01        Logic circuit assembler
volume1/passm/part02
volume1/leolife             Game of Life


From -2413899175261208451
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,dd06a22f13e8570a,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-02-24 09:21:52 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i013: shar -- Pack and unpack shell archives
Keywords: shar
Message-ID: <6458@saturn.ucsc.edu>
Date: 24 Feb 89 17:21:52 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 694
Approved: koreth@ssyx.ucsc.edu

Submitted-by: John.Altstadt@moliman.molienergy.bc.ca (John Altstadt)
Posting-number: Volume 2, Issue 13
Archive-name: shar

[I know, this is like posting a uuencoded binary for "uudecode", but... -sg]

Here is a somewhat updated version of shar (including unshar) that was
posted to the net many years ago.  I have included the original readme
file (complete with typo's) plus many typo's of my own.  If somebody
wants to clean up the code, or add more features, be my guest.  Binaries
have been sent to the binaries group.

#    This is a shell archive.
#    Remove everything above and including the cut line.
#    Then run the rest of the file through sh.
#----cut here-----cut here-----cut here-----cut here----#
#!/bin/sh
# shar:  Shell Archiver
#    Run the following text with /bin/sh to create:
#    readme
#    makefile
#    shar.c
# This archive created: 31-January-1989 20:09:20 ENOENV
```

```
# By:     ENOENV (ENOENV)
echo shar: extracting readme
sed 's/^X//' << \SHAR_EOF > readme
X
XThe next two files make up a shar/unshare program for the ST and vax VMS.  I
Xhave tested both versions and they work the same on each system.  I've had
Xthis source for quite a while and have been using it on VMS, but hadn't
Xbothered to port it to the ST until recently.  The first file is the shar
Xarchive for source and the link68 linkfile.  The second file is the uuencoded
Xbinary shar.prg.  I haven't prefixed the lines so that you can easily
Xextract the files without the progrm itself.  The original was posted to
Xnet.sources last year.  This one has had some bugs removed and most of the
Xamiga/cpm stuff deleted (it made it difficult to decipher the code).  But I
Xleft the VMS conditionals in place to make it easy to get versions running
Xon the two machines.  Hope you can find a use for this.  I sure have.
X
SHAR_EOF
if test 819 -ne "`wc -c readme`"
then
echo shar: error transmitting readme '(should have been 819 characters)'
fi
echo shar: extracting makefile
sed 's/^X//' << \SHAR_EOF > makefile
XCFLAGS  = -A -VSMALL
X
Xshar.ttp : shar.c makefile
X    cc shar.c $(CFLAGS) -o shar.ttp
X    strip shar.ttp
SHAR_EOF
if test 97 -ne "`wc -c makefile`"
then
echo shar: error transmitting makefile '(should have been 97 characters)'
fi
echo shar: extracting shar.c
sed 's/^X//' << \SHAR_EOF > shar.c
X/*
X *  Shar puts readable text files together in a package
X *  from which they are easy to extract.
X *
X *  v 860716 M. Kersenbrock (tektronix!copper!michaelk) for Z80-CPM
X *  - enhanced usage message
X *
X *  v 860712 D. Wecker for ULTRIX and the AMIGA
X *  - stripped down.. does patterns but no directories
X *  - added a -u (unshar) switch
X *
X *  v 870128 R. Royar for ST
X *  - removed #ifdefs that left stray }s in file
X *  - reformatted text for consistant style
X *  - modified main to abort when no files are given
X *  - fixed call to fputc in tell macro, which omitted the stream
X *    causing address errors on the ST.
X *  - fixed STUPID coding in getopt that tried to write to
```

```
X *    address 0++.
X *  - known bugs:
X *       still no directory support.  Extracting
X *       files with directory names will get
X *       only the directory name on an open call.
X *
X *  v 880713 J. Altstadt for ST
X *  - removed extraneous #ifdefs for all other systems
X *  - reformatted text for consistant style
X *  - changed lots of things for unsharing:
X *    1. added directory handling
X *    2. added code to handle weird shar sed formats better,
X *       will add more things when i get files that i can't unshar,
X *       eg. when i get one of those icky sharchives that has split files
X *       but won't add in the ability to handle shars that use awk to
X *       extract files (yes, windsrch uses awk, blech!!!)
X *    3. added in file overwrite protection, i didn't need it, but it was
X *       easy (and i know i will need it some day when i don't have a
X *       backup)
X *    4. skip whitespace at begining of lines intended for sh, so it will
X *       do stuff embedded in "if test" structures
X *
X *  v 890124 J. Altstadt for ST
X *  - unshar will handle files split across two (or more) sharchives
X *  - fixed bug that would allow creation of directories (files) with
X *    the same name as an existing file (directory)
X */
X
X
X#include <stdio.h>
X#include <osbind.h>
X#include <ctype.h>
X#include <stat.h>
X
Xlong _stksize = 16 * 1024; /* for 10k stack */
Xchar *Defenv = "ENOENV";   /* if no cc.ini file */
Xextern char *getenv(),*malloc(),*index();
Xvoid shar();
X
X
X#define BADCH     ((int)'?')
X#define EMSG ""
X#define tell(s) {fputs(*nargv,stderr);fputs((s),stderr);fputc('\n',stderr);}
X#define rescanopts()  (optind = 1)
X
Xint optind = 1,   /* index into parent argv vector */
X    optopt;      /* character checked for validity */
Xlong    fsize;        /* length of file */
Xchar    *optarg; /* argument associated with option */
Xchar    *sav[100];    /* saved file names */
Xint savind;       /* save index */
X
X/* OPTIONS */
```

```
Xint Verbose = 0;       /* provide append/extract feedback */
Xint Basename = 0;      /* extract into basenames */
Xint Count = 0;         /* count characters to check transfer */
Xchar    *Delim = "SHAR_EOF";    /* put after each file */
Xchar    Filter[100] = "cat";    /* used to extract archived files */
Xchar    *Prefix = NULL;         /* line prefix to avoid funny chars */
Xint OverWrite = 0;          /* do we overwrite an output file? */
X
Xchar Usage1[] =
X"\nSHAR: Create/extract file archive for extraction by /bin/sh (normally).\n\
X\n\
Xusage: shar [[-uo] archive] [[-a] [-p prefix]\
X [-d delim] [-bcv] files > archive]\n\
X\n\
X    where:    -a   all the options (v,c,b,-pXX)\n";
X
Xchar Usage2[] =
X"        -b   extract absolute paths into current directory\n\
X        -c   check filesizes on extraction\n\
X        -d   use this EOF delimiter instead of SHAR_EOF\n";
X
Xchar Usage3[] =
X"        -o   unshar <archive>, overwriting old files\n\
X        -p   use this as prefix to each line in archived files\n\
X        -u   unshar <archive>, not overwriting old files\n\
X        -v   verbose on extraction, incl. echoing filesizes\n";
X
X
X#define SED "sed 's/^%s//'"     /* used to remove prefix from lines */
X#define OPTSTRING "u:o:ap:d:bcv"
X
X
Xvoid main(argc, argv)
Xint argc;
Xchar **argv;
X{
X    char *ppchFiles[256];
X    register int  C;
X    char **ppchList = ppchFiles;
X    register int errflg;
X    errflg = 0;
X
X    while(EOF != (C = getopt(argc, argv, OPTSTRING))) {
X        switch(C) {
X        case 'v':
X            Verbose++;
X            break;
X        case 'c':
X            Count++;
X            break;
X        case 'b':
X            Basename++;
X            break;
```

```
X        case 'd':
X             Delim = optarg;
X             break;
X        case 'a': /* all the options */
X             optarg = "X";
X             Verbose++;
X             Count++;
X             Basename++;
X             /* fall through to set prefix */
X        case 'p':
X             sprintf(Filter, SED, Prefix = optarg);
X             break;
X        case 'o':
X             OverWrite++;
X        case 'u':
X             dounshar(optarg);
X             exit(0);
X             break;
X        default:
X             errflg++;
X        }
X    }
X
X    C = getarg(argc, argv);
X    if (errflg || EOF == C) {
X        if (EOF == C)
X             fprintf(stderr, "shar: No input files\n");
X        fprintf(stderr, "%s%s%s", Usage1, Usage2, Usage3);
X        exit(1);
X    }
X    savind = 0;
X
X    do {
X        if (getpat(optarg)) exit(2);
X    } while (EOF != (C = getarg(argc, argv)));
X
X    sav[savind] = 0;
X    header(sav);
X    for (ppchList = sav; *ppchList; ++ppchList)
X        shar(*ppchList);
X    puts("#\tEnd of shell archive");
X    puts("exit 0");
X    exit(0);
X}
X
X
Xint header(ppchFiles)
Xchar *ppchFiles[];
X{
X    char clock[40];
X    register char **ppchList;
X    char *pchOrg;
X    char *pchName;
```

```
X    register int  problems;
X    problems = 0;
X    pchOrg = getenv("organization");
X    pchName = getenv("name");
X    puts("#\tThis is a shell archive.");
X    puts("#\tRemove everything above and including the cut line.");
X    puts("#\tThen run the rest of the file through sh.");
X    puts("#----cut here-----cut here-----cut here-----cut here----#");
X    puts("#!/bin/sh");
X    puts("# shar: Shell Archiver");
X    puts("#\tRun the following text with /bin/sh to create:");
X    for (ppchList = ppchFiles; *ppchList; ++ppchList)
X         printf("#\t%s\n", *ppchList);
X    stime(clock);
X    printf("# This archive created: %s",clock);
X    if (pchName)
X         printf("# By:\t%s (%s)\n", pchName,pchOrg);
X    return(0);
X}
X
X
X/* create an ASCII time string and copy it into timeline.  Caller
X * should make sure timeline has ~35 characters space.
X */
Xstime(timeline)
Xchar timeline[];
X{
X    register int mytime;
X    register int date;
X    register int hours, minutes, seconds;
X    int day, month, year;
X    static char *months[] = {"January","February","March","April",
X                  "May","June","July","August","September",
X                  "October","November","December"};
X    char *tzone;
X    tzone = getenv("timezone");
X    mytime   = Tgettime();
X    date = Tgetdate();
X    seconds = (mytime&31)*2;
X    minutes = (mytime>>5)&63;
X    hours    = (mytime>>11)&31;
X    day = (date&31);
X    month = ((date>>5)&15)-1;
X    year = ((date>>9)&31)+1980;
X    sprintf(timeline,"%02d-%s-%d %d:%02d:%02d %s\n",
X    day,months[month],year,hours,minutes,seconds,tzone);
X    return(1);
X}
X
X
Xint
Xarchive(input, output)
Xchar *input, *output;
```

```
X{
X    char line[BUFSIZ];
X    register FILE *ioptr;
X
X    if (ioptr = fopen(input, "r")) {
X         printf("%s << \\%s > %s\n", Filter, Delim, output);
X         while(fgets(line, BUFSIZ, ioptr)) {
X              if (Prefix)
X                   fputs(Prefix, stdout);
X              fputs(line, stdout);
X              if (Count)
X                   fsize += strlen(line);
X         }
X         puts(Delim);
X         fclose(ioptr);
X         return(0);
X    }
X    else {
X         fprintf(stderr, "shar: Can't open '%s'\n", input);
X         return(1);
X    }
X}
X
X
Xvoid shar(file)
Xchar *file;
X{
X    register char *basefile;
X    basefile = file;
X    if (!strcmp(file, "."))
X         return;
X    fsize = 0;
X    if (Basename) {
X         while(*basefile)
X              basefile++;        /* go to end of name */
X         while(basefile > file && *(basefile-1) != '/')
X              basefile--;
X    }
X    if (Verbose)
X         printf("echo shar: extracting %s\n", basefile);
X    if (archive(file, basefile))
X         exit(66);
X    if (Count) {
X         printf("if test %ld -ne \"`wc -c %s`\"\n",fsize,basefile);
X         printf("then\necho shar: error transmitting %s ",basefile);
X         printf("'(should have been %ld characters)'\nfi\n",fsize);
X    }
X}
X
X
Xgetpat(pattern)
Xchar *pattern;
X{
```

```
X    register char *ptr;
X
X    ptr = pattern;
X    sav[savind] = malloc(strlen(ptr)+1);
X    strcpy(sav[savind++],ptr);
X    if (access(ptr,4)) {
X         printf("No read access for file: %s\n",ptr);
X         return(-1);
X    }
X    return(0);
X}
X
X
X/*
X * get option letter from argument vector
X */
Xint
Xgetopt(nargc, nargv, ostr)
Xint nargc;
Xchar **nargv, *ostr;
X{
X    register char *oli;         /* option letter list index */
X    static char   *place = EMSG;     /* option letter processing */
X
X    if(!*place) {          /* update scanning pointer */
X         if(optind >= nargc || *(place = nargv[optind])
X                   != '-' || !*++place)
X              return(EOF);
X         if (*place == '-') {    /* found "--" */
X              ++optind;
X              return(EOF);
X         }
X    }    /* option letter okay? */
X    if ((optopt = (int)*place++) == (int)':'
X              || !(oli = index(ostr,optopt))) {
X         if(!*place)
X              ++optind;
X         tell(": illegal option");
X    }
X    if (oli) {         /* %$^& what if oli = NULL!!! */
X         if (*++oli != ':') {        /* don't need argument */
X              optarg = NULL;
X              if (!*place)
X                   ++optind;
X         }
X         else {                  /* need an argument */
X              if (*place)        /* no white space */
X                   optarg = place;
X              else {
X                   if (nargc <= ++optind) {    /* no arg */
X                        place = EMSG;
X                        tell(": option requires an argument");
X                   }
```

```
X                  else
X                       optarg = nargv[optind]; /*white space*/
X             }
X             place = EMSG;
X             ++optind;
X         }
X    }
X    return(optopt);          /* dump back option letter */
X}
X
X
Xint
Xgetarg(nargc, nargv)
Xint nargc;
Xchar **nargv;
X{
X    if (nargc <= optind) {
X         optarg = (char *) 0;
X         return(EOF);
X    }
X    else {
X         optarg = nargv[optind++];
X         return 0;
X    }
X}
X
X
Xdounshar(ArcNam)
Xchar *ArcNam;
X{
X    register int i;
X    register FILE *inptr,*outptr;
X    register int hundreds;
X    char line[BUFSIZ];
X    int DirNum = -1;
X    int Prefix = 0;
X    char Dirs[10][40], FilNam[428], Delim[128], ScrStr[28];
X    char strip[40];
X    char *ptr, *lptr;
X    int wipeold, append, exists;
X    struct stat dirstat;
X
X    if (!(inptr = fopen(ArcNam,"r"))) {
X         fprintf(stderr,"shar: Can't open archive '%s'\n", ArcNam);
X         return;
X    }
X    while (fgets(line, BUFSIZ, inptr)) {
X         for (lptr = line; (*lptr != (char) 0) && (isspace(*lptr));
X                   lptr++)
X              ;
X         if (strncmp(lptr, "sed ", 4) == 0) {
X              Prefix = 0;
X              strcpy(strip, "");
```

```
X             if (!(ptr = index(lptr,'/')))
X                  goto getfil;
X             if (*++ptr == '^')
X                  ++ptr;
X             while (*ptr != '/') {
X                  strncat(strip, ptr, 1);
X                  Prefix++;
X                  *(strip + Prefix) = (char) 0;
X                  ptr++;
X             }
X             goto getfil;
X        }
X        else if (strncmp(lptr, "gres ", 5) == 0) {
X             Prefix = 0;
X             strcpy(strip, "");
X             if (!(ptr = index(lptr,'\'')))
X                  goto getfil;
X             if (*++ptr == '^')
X                  ++ptr;
X             while (*ptr != '\'') {
X                  if (*ptr == '0')
X                       continue;
X                  strncat(strip, ptr, 1);
X                  Prefix++;
X                  *(strip + Prefix) = (char) 0;
X                  ptr++;
X             }
X             goto getfil;
X        }
X        else if (strncmp(lptr,"cat ",4) == 0) {
X             Prefix = 0;
Xgetfil:
X             FilNam[0] = 0;
X
X             for (i = 0; i <= DirNum; i++) {
X                  strcat(FilNam,Dirs[i]);
X                  strcat(FilNam,"\\");
X             }
X
X             append = 0;
X             getshpar(lptr, ">", ScrStr, 1);
X             if (strcmp(">", ScrStr) == 0) {
X                  getshpar(lptr, ">>", ScrStr, 1);
X                  append = 1;
X             }
X             strcat(FilNam,ScrStr);
X             getshpar(lptr, "<<", Delim, 0);
X             wipeold = 1;
X             exists = 1;
X
X             i = stat(FilNam, &dirstat);
X             i = i < 0 ? i : (int) dirstat.st_mode;
X             if ((i >= 0) && (i <= 0X3F)) {
```

```
X                 if (!OverWrite)
X                      wipeold = 0;
X             }
X             else
X                  exists = 0;
X
X             if (wipeold && !append) {
X                  outptr = fopen(FilNam, "w");
X                  fprintf(stderr, "Creating %s ", FilNam);
X             }
X             else if (append && exists) {
X                  outptr = fopen(FilNam, "a");
X                  fprintf(stderr, "Appending %s ", FilNam);
X             }
X             else if (append && !exists) {
X                  fprintf(stderr, "Cannot append to %s,\n\
X\tyou need to unshar another sharchive first ", FilNam);
X             }
X             else
X                  fprintf(stderr, "Will not overwrite %s ",
X                            FilNam);
X             if (!outptr) {
X                  fprintf(stderr, "Fatal:  could not open file \
X%s\n", FilNam);
X                  exit(1);
X             }
X             fflush(stderr);
X             hundreds = 0;
X             while (fgets(line, BUFSIZ, inptr)) {
X                  if (strncmp(line, Delim, strlen(Delim)) == 0)
X                       break;
X                  hundreds++;
X                  if (!(hundreds % 100)) {
X                       fprintf(stderr, ".");
X                       fflush(stderr);
X                  }
X                  if ((wipeold && !append) || (append && exists))
X                       if (strncmp(line, strip, Prefix))
X                            fputs(line, outptr);
X                       else
X                            fputs(&line[Prefix],outptr);
X             }
X             if (outptr) {
X                  fclose(outptr);
X                  fprintf(stderr,"done\n");
X             }
X             else {
X                  fprintf(stderr," error in creating file\n");
X                  exit(1);
X             }
X        }
X        else if (strncmp(lptr, "mkdir ", 6) == 0) {
X             FilNam[0] = 0;
```

```
X
X             for (i = 0; i <= DirNum; i++) {
X                  strcat(FilNam,Dirs[i]);
X                  strcat(FilNam,"\\");
X             }
X
X             getshpar(lptr, "mkdir", ScrStr, 1);
X             strcat(FilNam, ScrStr);
X
X             i = stat(FilNam, &dirstat);
X             i = i < 0 ? i : (int) dirstat.st_mode;
X             if ((i >= 0) && (i <= 0X3F))
X                  if ((i & 0X10) == 0X10)
X                       continue; /* don't re-create */
X                  else {
X                       fprintf(stderr, "will not mkdir %s,\n\
X\tthere is already a file of that name\n", FilNam);
X                       exit(1);
X                  }
X             if (Dcreate(FilNam)) {
X                  fprintf(stderr, "could not mkdir %s\n", FilNam);
X                  exit(1);
X             }
X        }
X        else if (strncmp(lptr,"cd ",3) == 0) {
X             if (lptr[3] == '.' && lptr[4] == '.')
X                  DirNum--;
X             else if (DirNum >= 9)
X                  fprintf(stderr,"directories nested too deep\n");
X             else {
X                  DirNum++;
X                  getshpar(lptr, "cd", ScrStr, 1);
X                  strcpy(Dirs[DirNum], ScrStr);
X             }
X             if (DirNum < -1)
X                  DirNum = -1;
X        }
X        else if (strncmp(lptr, "chdir ", 6) == 0) {
X             if (lptr[6] == '.' && lptr[7] == '.')
X                  DirNum--;
X             else if (DirNum >= 9)
X                  fprintf(stderr,"directories nested too deep\n");
X             else {
X                  DirNum++;
X                  getshpar(lptr, "cd", ScrStr, 1);
X                  strcpy(Dirs[DirNum], ScrStr);
X             }
X             if (DirNum < -1)
X                  DirNum = -1;
X        }
X    }
X    fclose(inptr);
X}
```

```
X
X
Xgetshpar(line, sea, par, flipslash)
Xchar *line, *sea, *par;
Xint flipslash;
X{
X    register char *scr1, *scr2;
X
X    while (*line) {
X         scr1 = line;
X         scr2 = sea;
X         while (*scr1 && *scr2 && *scr1 == *scr2) {
X              scr1++;
X              scr2++;
X         }
X         if (*scr2 == 0) {
X              if (*scr1 == 0) {
X                   *par = 0;
X                   return;
X              }
X              while (*scr1 == ' ' || *scr1 == '\t' ||
X                        *scr1 == '\\' || *scr1 == '\'' ||
X                        *scr1 == '"')
X                   scr1++;
X              while (*scr1 != 0 && *scr1 != ' ' && *scr1 != '\t' &&
X                        *scr1 != '\\' && *scr1 != '\'' &&
X                        *scr1 != '"' && *scr1 != '\n' &&
X                        *scr1 != '\r') {
X                   if ((*scr1 == '/') && flipslash) {
X                        *par++ = '\\';
X                        scr1++;
X                   }
X                   else
X                        *par++ = *scr1++;
X              }
X              if (*(par - 1) == '\\')/* if \ at end, not ST path */
X                   par--;
X              *par = 0;
X              return;
X         }
X         line++;
X    }
X    *par = 0;
X}
X
X
X/* since nothing ever initializes the environment table on the ST
X * I'll just use my home brew cc.ini environment file.
X */
Xchar *getenv(var)
Xregister char *var;
X{
X    FILE *fp;
```

```
X    register char *ptr;
X    static char line[81];
X    register char *safestore;
X
X    if ((fp=fopen("cc.ini","r")) == NULL)
X         return(Defenv);
X    while (fgets(line,80,fp)) {
X         if ((ptr=index(line,' '))!=(char *)NULL)
X              *ptr = '\0';
X         if ((ptr=index(line,'\n'))!=(char *)NULL)
X              *ptr = '\0';
X         if ((ptr=index(line,'\t'))!=(char *)NULL)
X              *ptr = '\0';
X         if ((ptr=index(line,'='))!=(char *)NULL)
X              *ptr = '\0';
X         else
X              continue;
X         ++ptr;
X         if (strcmp(line, var) == 0) {
X              safestore = malloc(strlen(ptr)+1);
X              strcpy(safestore,ptr);
X              return(safestore);
X         }
X         else if (feof(fp))
X              break;
X    }
X    return(Defenv);
X}
SHAR_EOF
if test 14606 -ne "`wc -c shar.c`"
then
echo shar: error transmitting shar.c '(should have been 14606 characters)'
fi
#    End of shell archive
exit 0


From -4128375310609496775
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,1186e62114e7c79f,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-02-24 09:32:07 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i014: keyfix -- Fix Gulam for non-American keyboards
Keywords: C
Message-ID: <6460@saturn.ucsc.edu>
Date: 24 Feb 89 17:32:07 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 332
Approved: koreth@ssyx.ucsc.edu
```

Submitted-by: uunet!mcvax!cwi.nl!eddy (Eddy Boeve)
Posting-number: Volume 2, Issue 14
Archive-name: keyfix

Here is a program to fix the 'gulam.prg' binary for non-american keyboards.
You do not have to edit the program, just run it. The program does the
rest. The binary is also being posted in comp.binaries.atari.st.

```
/*
 *   keyfix.c -- Utility for Gulam
 *
 *   Original version: copyright (c) 1986, 1987 pm@cwru.edu
 *
 *   Rewrite  : eddyright (e) 1989 eddy@cwi.nl
 *
 *   This is an utility for GULAM to patch the default key bindings
 *   in the 'gulam.prg' binary. It was originally created by
 *   'pm@cwru.edu' to change the USA key bindings for us Europeans
 *   and contained a 3*128 table with the desired new bindings.
 *
 *   This version is different because you don't have to edit it for
 *   your own machine. It works out for itself what keys produce what
 *   characters.
 *
 *   This program takes your default key translation tables via the
 *   XBIOS function 'Keytbl' and adds the GULAM function, arrow and
 *   keypad key translations. (Why Gulam doesn't do this is unclear).
 *   If you already have a USA key binding, everything will stay the
 *   same. Otherwise the '@~#\|$' keys will hopefully appear under the
 *   right button.
 *
 *   Compile 'kmpfix.c'.  Run the resulting program from the directory
 *   that contains the 'gulam.prg'.
 *   The new version of 'gulam.prg' (with the new key mapping) will be
 *   written out as 'gulamnew.prg' in the current directory.
 *
 *   Below we define ST520/1040 keynames; the values are the ascii codes
 *   returned by ST BIOS.
 *
 *   The values below are chosen to have certain patterns:
 *
 *   to SHIFT   add 0040
 *   F1-10           0201 to 0212
 *   keypad 0-9      0260 to 0272 (subtract -0200 to get ascii digits)
 *   keypad()/ *-+.cr  add 0300 to the ascii ()/ *-+.cr
 *
 *   Other choices may be better, I don't know.  However, I think it is
 *   unwise to use numbers in the range 0000 to 0177, because they are
 *   regular ascii codes.
 *
 *   If you want to change some bindings, do it in the routine
 *   'copy_map':
```

```
 *       kn['scan_code'] = 'ascii_code';
 *
 *   Exit values:
 *        0   : No errors,
 *       -1   : Error in finding key map in binary,
 *       -2   : Error in memory allocation,
 *       -3   : Error in file reading/writing.
 */

#include <osbind.h>
#include <xbios.h>
#include <bios.h>

#define GULAM_SIZE (100000L) /* Number larger than size gulam
                        * program (in bytes) */

/*
 *   These 'KC_' scan code values are not defined in <bios.h>
 */
#define  KC_KLP        0x63 /* Keypad (   */
#define  KC_KRP        0x64 /* Keypad )   */
#define  KC_KSTAR  0x66 /* Keypad *   */
#define  KC_KPLUS  0x4E /* Keypad +   */
#define  KC_KENT       0x72 /* Keypad Enter    */
#define  KC_KMINUS 0x4A /* Keypad -   */
#define  KC_KDOT       0x71 /* Keypad .   */
#define  KC_KSLASH 0x65 /* Keypad /   */
#define  KC_K0         0x70 /* Keypad 0   */
#define  KC_K1         0x6D /* Keypad 1   */
#define  KC_K2         0x6E /* Keypad 2   */
#define  KC_K3         0x6F /* Keypad 3   */
#define  KC_K4         0x6A /* Keypad 4   */
#define  KC_K5         0x6B /* Keypad 5   */
#define  KC_K6         0x6C /* Keypad 6   */
#define  KC_K7         0x67 /* Keypad 7   */
#define  KC_K8         0x68 /* Keypad 8   */
#define  KC_K9         0x69 /* Keypad 9   */

/*
 *   ST520/1040 keynames
 */
#define F1    0201
#define F2    0202
#define  F3   0203
#define F4    0204
#define F5    0205
#define F6      0206
#define F7    0207
#define F8    0210
#define F9    0211
#define F10   0212

#define HELP    0213
```

```
#define UNDO  0214
#define INSERT    0215
#define HOME  0216

#define UPARRO    0273
#define DNARRO    0274
#define LTARRO    0275
#define RTARRO    0276

#define  KLP  0300
#define  KRP  0301
#define  KSTAR    0302
#define  KPLUS    0303
#define  KENTER   0304
#define  KMINUS   0305
#define  KDOT 0306
#define  KSLASH   0307

#define  K0   0260
#define  K1   0261
#define  K2   0262
#define  K3   0263
#define  K4   0264
#define  K5   0265
#define  K6   0266
#define  K7   0267
#define  K8   0270
#define  K9   0271


/*
 *   The above are the unshifted values.  Shifted, as well as
 *   'capslocked' (for only these keys) values that I use are these +0040.
 */
#define SHIFTED 0040


#define UNSHIFTED_MODE  0    /* Normal (unshifted) mode  */
#define SHIFTED_MODE    1    /* With shift key pressed   */
#define CAPSLOCK_MODE   2    /* With capslock key pressed
     */

static
char *mode_string [] =
     {
          "Unshifted Mode",  /* Mode_string[UNSHIFTED_MODE]
     */
          "Shifted Mode",    /* Mode_string[SHIFTED_MODE]
     */
          "Capslock Mode"    /* Mode_string[CAPSLOCK_MODE]
     */
     };
```

```
/*
 *   Error ( string, ec)
 *
 *   Write string and exit (ec).
 */
Error (str, ec)
char *str;
int ec;
{
     printf ("Error: ** %s **\n", str);
     exit (ec);

} /* error */

/*
 *   copy_map (mode, k_orig, k_new)
 *
 *   This routine will copy the contents of 'k_orig' (the original
 *   keyboard's translation table for mode 'mode', obtained by the
 *   xbios function 'Keytbl') to the new defined translation table
 *   'k_new'. Also the function/arrow/keypad key entries in 'k_new'
 *   will be changed to the preferred Gulam settings.
 */
void copy_map (mode, k_orig, k_new)
register short  mode;
char *k_orig,  /* Pointer to original key table for 'mode' */
     *k_new;       /* New defined key table for Gulam */
{
     register char *ko = k_orig,
                   *kn = k_new;
     register int  i,
                   offset = 0;   /* If shifted mode, add this */

     printf ("Changing keyboard translation table (%s) at %06lx.\n",
          mode_string[mode], k_orig);

     for (i=0; i<128; i++)
          kn[i] = ko[i];

     /*
      *   Change Function/Arrow/Keypad keys
      *   Here you can add/change entries.
         */
     if (mode == SHIFTED_MODE)
          offset = SHIFTED;

     kn[KC_F1] = F1 + offset;
     kn[KC_F2] = F2 + offset;
     kn[KC_F3] = F3 + offset;
     kn[KC_F4] = F4 + offset;
     kn[KC_F5] = F5 + offset;
     kn[KC_F6] = F6 + offset;
     kn[KC_F7] = F7 + offset;
```

```
     kn[KC_F8] = F8 + offset;
     kn[KC_F9] = F9 + offset;
     kn[KC_F10]    = F10 + offset;

     kn[KC_HELP]   = HELP + offset;
     kn[KC_UNDO]   = UNDO + offset;
     kn[KC_INS]    = INSERT + offset;
     kn[KC_CLR]    = HOME + offset;

     kn[KC_CUP]    = UPARRO + offset;
     kn[KC_CDOWN]  = DNARRO + offset;
     kn[KC_CLEFT]  = LTARRO + offset;
     kn[KC_CRIGHT] = RTARRO + offset;

     kn[KC_KLP]    = KLP + offset;
     kn[KC_KRP]    = KRP + offset;
     kn[KC_KSLASH] = KSLASH + offset;
     kn[KC_KSTAR]  = KSTAR + offset;
     kn[KC_KPLUS]  = KPLUS + offset;
     kn[KC_KENT]   = KENTER + offset;
     kn[KC_KMINUS] = KMINUS + offset;
     kn[KC_KDOT]   = KDOT + offset;
     kn[KC_K0] = K0 + offset;
     kn[KC_K1] = K1 + offset;
     kn[KC_K2] = K2 + offset;
     kn[KC_K3] = K3 + offset;
     kn[KC_K4] = K4 + offset;
     kn[KC_K5] = K5 + offset;
     kn[KC_K6] = K6 + offset;
     kn[KC_K7] = K7 + offset;
     kn[KC_K8] = K8 + offset;
     kn[KC_K9] = K9 + offset;

} /* copy_map */


/*
 *   Return offset in the binary file gulam.prg where the keymap[] begins.
 *      80336 (decimal) is this number valid for beta-version
 *   1.03.04.05 (121887).
 */

char * kmpbgn(buf, n)
register char *buf;
register long n;
{
     /* kmp[] is extracted from Gulam's src; do NOT change it. */
     static char   kmp[16] = {
          /* -------scan codes: 0x00-0x0f --------------- */
          '\000' /*nul*/, '\033' /*esc*/,
          '1' ,    '2' ,    '3' , '4' ,   '5' ,    '6' ,
          '7' ,    '8' ,    '9' , '0' ,   '-' ,    '=' ,
          '\010' /* bs*/, '\011' /* ht*/
```

```
    };

    register long i;
    register int  j;
    register char *p, *q;

    for (i = 0L; i < n; i++)
    {    p = &buf[i]; q = kmp;
         for (j = 0; j < 16; j++)
              if (*p++ != *q++) break;
         if (j == 16) goto found;
    }
    Error ("key map begin not found!", -1);

    found:   printf("Key map begins at byte %ld\r\n", i);
    return   &buf[i];
}


main()
{
    struct keytbl *kp;
    register char *buf, *q;
    register int  i;
    register long n, m;
    register unsigned long lsz;

    lsz  = GULAM_SIZE;     /* Enough space to hold entire
                       * gulam.prg! */

    buf = (char *) Malloc(lsz);
        if (buf == 0L)
         Error ("can't Malloc", -2);

    /* read all of gulam.prg into buf */
    i = Fopen("gulam.prg", 0);
        if (i <= 0)
         Error ("gulam.prg not found", -3);
    n =  Fread(i, lsz, buf);
        if (n <= 0L)
         Error ("error reading file", -3);
        else
         if (n == GULAM_SIZE)
             Error ("gulam.prg probably bigger", -3);
       Fclose(i);

    /*
     *   Get the keymap beginning in 'gulam.prg'
     */
    q = kmpbgn(buf, n);

    /*
     *   Fix the keymap tables
```

```
     */
    kp = Keytbl (-1L,-1L,-1L);       /* Get keytable structure */
    copy_map (UNSHIFTED_MODE, kp->kt_normal,   (char *) q );
    copy_map (SHIFTED_MODE,   kp->kt_shifted,  (char *) (q + 128) );
    copy_map (CAPSLOCK_MODE,  kp->kt_capslock, (char *) (q + 256) );

    /* write the new gulam.prg out */
    i = Fcreate("gulamnew.prg", 0);
        if (i <= 0)
         Error ("could not create gulamnew.prg", -3);

    m =  Fwrite(i, n, buf);
        if (m != n)
         Error ("error writing file", -3);
       Fclose(i);

    printf("new version of gulam is gulamnew.prg\n\n");

} /* main */
```

```
From -8811237670534869751
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,f8b74737f555003,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-02-24 09:37:22 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i015: sed -- GNU sed (stream editor)  part01/02
Keywords: shar
Message-ID: <6461@saturn.ucsc.edu>
Date: 24 Feb 89 17:37:22 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1358
Approved: koreth@ssyx.ucsc.edu

Submitted-by: saj@chinet.chi.il.us (Stephen Jacobs)
Posting-number: Volume 2, Issue 15
Archive-name: sed/part01

This is the first of 2 parts of source/text and one part of binary of sed
working on the Atari ST.  The executable was compiled with Mark Williams
C, v3.06.  Due to the limitations on distribution mentioned in sed.man,
I think sed.man should not be arc-ed together with the executable, but should
only be part of the source distribution.  [That's what I've done. -sg]

It is an implementation of the Unix utility of the same name, and that even
though it's named sed.ttp it really shouldn't be run from the desktop
because the input is case sensitive.

#! /bin/sh
```

```
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of shell archive."
# Contents:  ctrans debug.h pascal sajnote.sed sed.h sed.man sedcomp.c
# Wrapped by saj@chinet on Sun Jan 29 23:52:01 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f ctrans -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"ctrans\"
else
echo shar: Extracting \"ctrans\" \(1134 characters\)
sed "s/^X//" >ctrans <<'END_OF_ctrans'
X# First, change over comment delimiters
X        /{/s//\/* /g
X        /}/s// *\//g
X# Then the block start and end
X        /begin/s//{/g
X        /end/s//}/g
X# Stash away assignment ops and the relationals with = in them
X        /:=/s//::/g
X        />=/s//>:/g
X        /<=/s//<:/g
X# Remaining Pascal = are C ==
X        /=/s//==/g
X# Now convert to C assignment syntax and restore relationals
X        /::/s//=/g
X        />:/s//>=/g
X        /<:/s//<=/g
X# Now convert Pascal's not-equal
X        /<>/s//!=/g
X# Eliminate unused thens
X        / then/s///g
X# Convert to C logical operators
X        / or /s// || /g
X        / and /s// && /g
X# Convert modulo operator
X        / mod /s// % /g
X# Now convert procedure syntax
X        /procedure/s/);$/)/
X        /procedure/s//void/
X# So C won't think it's a declare
X        /function/s/function \(.*\): *\(.*\);/function \2 \1/
X        /function/s/: [a-zA-Z0-9]+//
X        /function +/s///
X# First hack at changing strings
X        /'/s//"/g
X# Turn space pairs to tabs
X        /^  /s/  /      /g
X# Change standard types
X        /integer/s//int/g
X        /integer/s//bool/g
```

```
X        /cycle/s//continue/g
END_OF_ctrans
if test 1134 -ne `wc -c <ctrans`; then
    echo shar: \"ctrans\" unpacked with wrong size!
fi
# end of overwriting check
fi
if test -f debug.h -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"debug.h\"
else
echo shar: Extracting \"debug.h\" \(348 characters\)
sed "s/^X//" >debug.h <<'END_OF_debug.h'
X#define PASS(msgstring)/* printf(msgstring), printf ("\n")*/
X
X/* This debug header was added by Stephen Jacobs, and isn't part of GNU
X*sed.  It may be hacked omitted or replaced at will.  The only other
X*thing I've done is fix a compiler dependency and an inappropriate declaration,
X*which are marked in the source files with my initials, SAJ.
X*/
END_OF_debug.h
if test 348 -ne `wc -c <debug.h`; then
    echo shar: \"debug.h\" unpacked with wrong size!
fi
# end of overwriting check
fi
if test -f pascal -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"pascal\"
else
echo shar: Extracting \"pascal\" \(111 characters\)
sed "s/^X//" >pascal <<'END_OF_pascal'
X{ this is a test }
Xx := y
Xbegin
Xend
Xif (x>=y)
Xx<>y
Xx or y
Xprocedure foo();
Xfunction fie(abc,def,gef): integer;
END_OF_pascal
if test 111 -ne `wc -c <pascal`; then
    echo shar: \"pascal\" unpacked with wrong size!
fi
# end of overwriting check
fi
if test -f sajnote.sed -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"sajnote.sed\"
else
echo shar: Extracting \"sajnote.sed\" \(1952 characters\)
sed "s/^X//" >sajnote.sed <<'END_OF_sajnote.sed'
XAfter putting an unreasonable amount of effort into debugging this beast, I
Xfeel entitled to add a few words to what's already in the .man file.
X
```

```
XThe main reason I wanted sed was to help distinguish defining and referencing
Xdeclarations of external variables in C.  Andy Tanenbaum got me started on
Xthis in his Operating Systems, Design and Implementation (the MINIX book).
XAnyway, I adopted his idea of a pre-processor macro that expands to a
Xgramatically proper referencing declaration everywhere except in one file,
Xwhere it is redefined to expand to a proper defining declaration.  This does
Xnot serve for initialized external variables, however.  My solution follows:
X
XThe macros EXTERN add Initialize are defined as
X
X#define EXTERN extern
X#define INITIALIZE(type, name, value) extern type name
X
Xexcept in one file, which says
X
X#undef EXTERN
X#undef INITIALIZE
X#define EXTERN
X#define INITIALIZE(type, name, value) type name = (value)
X
XThis allows things like
X
XINITIALIZE(double, imagintvl, -.0125);
XEXTERN double reintmp, imintmp;
XEXTERN struct complex upleft, center;
XINITIALIZE(int, drawarray[4], ({0, 9, 0, 0}));
X
XFine so far.  But that initializer {0. 9, 0, 0} really does need to be
Xenclosed in parentheses, because only parentheses can protect a comma from
Xbeing taken as a token delimiter by the C pre-processor.  Similarly,
Xthe INITIALIZE macro really should put the initializer in parentheses
Xto avoid unpleasant surprises.  Imagine my surprise when C compilers
Xrejected "int drawarray[4] = (({0, 9, 0, 0}));".  It seems that the syntax
Xof a multi-value initializer requires " = <optional whitespace> {".  Damn.
X
XSed to the rescue!! My makefile for a complex program now includes
X
X
Xexpanded.c:  defines.c
X    cc -E defines.c > $(TMPDIR)\expanded
X    sed -e '/=/s/(\([{"].*\));/\1;/' $(TMPDIR)\expanded > $(TMPDIR)\filtered
X    sed -e '/=/s/(\([{"].*\));/\1;/' $(TMPDIR)\filtered > expanded.c
X
XTa daaa!
X                  Stephen Jacobs (saj@chinet.chi.il.us)
END_OF_sajnote.sed
if test 1952 -ne `wc -c <sajnote.sed`; then
    echo shar: \"sajnote.sed\" unpacked with wrong size!
fi
# end of overwriting check
fi
if test -f sed.h -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"sed.h\"
```

```
else
echo shar: Extracting \"sed.h\" \(4015 characters\)
sed "s/^X//" >sed.h <<'END_OF_sed.h'
X/* sed.h -- types and constants for the stream editor */
X
X/* data area sizes used by both modules */
X#define MAXBUF          4000    /* current line buffer size */
X#define MAXAPPENDS      20      /* maximum number of appends */
X#define MAXTAGS         9       /* tagged patterns are \1 to \9 */
X
X/* constants for compiled-command representation */
X#define EQCMD   0x01    /* = -- print current line number              */
X#define ACMD    0x02    /* a -- append text after current line  */
X#define BCMD    0x03    /* b -- branch to label                        */
X#define CCMD    0x04    /* c -- change current line            */
X#define DCMD    0x05    /* d -- delete all of pattern space            */
X#define CDCMD   0x06    /* D -- delete first line of pattern space     */
X#define GCMD    0x07    /* g -- copy hold space to pattern space       */
X#define CGCMD   0x08    /* G -- append hold space to pattern space     */
X#define HCMD    0x09    /* h -- copy pattern space to hold space       */
X#define CHCMD   0x0A    /* H -- append hold space to pattern space     */
X#define ICMD    0x0B    /* i -- insert text before current line        */
X#define LCMD    0x0C    /* l -- print pattern space in escaped form    */
X#define NCMD    0x0D    /* n -- get next line into pattern space       */
X#define CNCMD   0x0E    /* N -- append next line to pattern space      */
X#define PCMD    0x0F    /* p -- print pattern space to output          */
X#define CPCMD   0x10    /* P -- print first line of pattern space      */
X#define QCMD    0x11    /* q -- exit the stream editor                 */
X#define RCMD    0x12    /* r -- read in a file after current line */
X#define SCMD    0x13    /* s -- regular-expression substitute          */
X#define TCMD    0x14    /* t -- branch on last substitute successful   */
X#define CTCMD   0x15    /* T -- branch on last substitute failed       */
X#define WCMD    0x16    /* w -- write pattern space to file            */
X#define CWCMD   0x17    /* W -- write first line of pattern space      */
X#define XCMD    0x18    /* x -- exhange pattern and hold spaces        */
X#define YCMD    0x19    /* y -- transliterate text                     */
X
Xstruct  cmd_t                          /* compiled-command representation */
X{
X        char    *addr1;                /* first address for command */
X        char    *addr2;                /* second address for command */
X        union
X        {
X                char            *lhs;   /* s command lhs */
X                struct cmd_t    *link;  /* label link */
X        } u;
X        char    command;               /* command code */
X        char    *rhs;                  /* s command replacement string */
X        FILE    *fout;                 /* associated output file descriptor */
X        struct
X        {
X                unsigned allbut  : 1;   /* was negation specified? */
X                unsigned global  : 1;   /* was p postfix specified? */
```

```
X                unsigned print   : 2;   /* was g postfix specified? */
X                unsigned inrange : 1;   /* in an address range? */
X        } flags;
X};
Xtypedef struct cmd_t    sedcmd;         /* use this name for declarations */
X
X#define BAD     ((char *) -1)           /* guaranteed not a string ptr */
X
X

X/* address and regular expression compiled-form markers */
X#define STAR    1       /* marker for Kleene star */
X#define CCHR    2       /* non-newline character to be matched follows */
X#define CDOT    4       /* dot wild-card marker */
X#define CCL     6       /* character class follows */
X#define CNL     8       /* match line start */
X#define CDOL    10      /* match line end */
X#define CBRA    12      /* tagged pattern start marker */
X#define CKET    14      /* tagged pattern end marker */
X#define CBACK   16      /* backslash-digit pair marker */
X#define CLNUM   18      /* numeric-address index follows */
X#define CEND    20      /* symbol for end-of-source */
X#define CEOF    22      /* end-of-field mark */
X
X/* sed.h ends here */
END_OF_sed.h
if test 4015 -ne `wc -c <sed.h`; then
    echo shar: \"sed.h\" unpacked with wrong size!
fi
# end of overwriting check
fi
if test -f sed.man -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"sed.man\"
else
echo shar: Extracting \"sed.man\" \(9360 characters\)
sed "s/^X//" >sed.man <<'END_OF_sed.man'
XNAME
X   sed - the stream editor
X
X
XSYNOPSIS
X   sed [-n] [-g] [-e script ] [-f sfile ] [ file ] ...
X
X
XDESCRIPTION
X   Sed copies the named files (standard input default) to the standard
Xoutput, edited according to a script of commands.
X   An -e option supplies a single edit command from the next argument;
Xif there are several of these they are executed in the order in  which
Xthey appear. If there is just one -e option and no -f 's, the -e  flag
Xmay be omitted.
X   An -f option causes commands to be taken from the file "sfile";  if
Xthere are several  of these they are  executed in the order  in  which
```

Xthey appear; -e and -f commands may be mixed.
X   The -g option causes sed to act as though every substitute command
Xin the script has a g suffix.
X   The -n option suppresses the default output.
X
X   A script consists of commands, one per line, of the following form:
X
X[address [, address]] function [arguments]
X
X   Normally sed cyclically copies a line of input into a current text
Xbuffer, then applies all commands whose addresses select the buffer in
Xsequence, then copies the buffer to standard output and clears it.
X   The -n option suppresses normal output (so that only p and w output
Xis done). Also, some commands (n, N) do their own line reads, and some
Xothers (d, D) cause all commands following in the script to be skipped
X(the D command also suppresses the clearing of the current text buffer
Xthat would normally occur before the next cycle).
X
X   It is also helpful to know that there's a second buffer (called the
X'hold space' that can be copied or appended to or from or swapped with
Xthe current text buffer.
X
X   An address is: a decimal numeral (which matches the line it numbers
Xwhere line numbers start at 1 and run cumulatively across files), or a
X`$' that addresses the last line of input, or a context address, which
Xis a `/regular expression/', in the style of ed (1) modified thus:
X
X(1) The escape sequence `\n' matches a newline embedded in the buffer,
X   and `\t' matches a tab.
X
X(2) A command line with no addresses selects every buffer.
X
X(3) A command line with one address selects every buffer that matches
X   that address.
X
X(4) A command line with two addresses selects the inclusive range from
X   the first input buffer that matches the first address through the
X   next input buffer that matches the second. (If the second address
X   is a number less than or equal to the line number first selected,
X   only one line is selected.) Once the second address is matched sed
X   starts looking for the first one again; thus, any number of these
X   ranges will be matched.
X
X   The negation operator '!' can prefix a command to apply it to every
Xline not selected by the address(es).
X
X   In the following list of functions, the maximum number of addresses
Xpermitted for each function is indicated in parentheses.
X   An argument denoted "text" consists of one or more lines, with all
Xbut the last ending with `\' to hide the newline.
X   Backslashes in text are treated like backslashes in the replacement
Xstring of an `s' command and may be used to protect initial whitespace
X(blanks and tabs) against the stripping that is done on every line of

Xthe script.
X
X   An argument denoted "rfile" or "wfile" must be last on the command
Xline. Each wfile is created before processing begins. There can be at
Xmost 10 distinct wfile arguments.
X
Xa "text"   (1)
X   Append. Place text on output before reading the next input line.
X
Xb "label" (2)
X   Branch to the `:' command bearing the label. If no label is given,
Xbranch to the end of the script.
X
Xc "text"   (2)
X   Change. Delete the current text buffer. With 0 or 1 address, or at
Xthe end of a 2-address range, place text on the output. Start the next
Xcycle.
X
Xd   (2)
X   Delete the current text buffer. Start the next cycle.
X
XD   (2)
X   Delete the first line of the current text buffer (all chars up to the
Xfirst newline). Start the next cycle.
X
Xg   (2)
X   Replace the contents of the current text buffer with the contents of
Xthe hold space.
X
XG   (2)
X   Append the contents of the hold space to the current text buffer.
X
Xh   (2)
X   Copy the current text buffer into the hold space.
X
XH   (2)
X   Append a copy of the current text buffer to the hold space.
X
Xi "text"   (1)
X   Insert. Place text on the standard output.
X
Xl   (2)
X   List. Sends the pattern space to standard output. A "w" option may
Xfollow as in the s command below. Non-printable characters expand to:
X   \b -- backspace (ASCII 08)
X   \t -- tab   (ASCII 09)
X   \n -- newline   (ASCII 10)
X   \r -- return   (ASCII 13)
X   \e -- escape   (ASCII 27)
X   \xx -- the ASCII character corresponding to 2 hex digits xx.
X
Xn   (2)
X   Copy the current text buffer to standard output. Read the next line

Xof input into it.
X
XN   (2)
X   Append the next line of input to the current text buffer, inserting
Xan embedded newline between the two. The current line number changes.
X
Xp   (2)
X   Print. Copy the current text buffer to the standard output.
X
XP   (2)
X   Copy the first line of the current text buffer (all chars up to the
Xfirst newline) to standard output.
X
Xq   (1)
X   Quit. Branch to the end of the script. Do not start a new cycle.
X
Xr "rfile" (1)
X   Read the contents of rfile. Place them on the output before reading
Xthe next input line.
X
Xs /regular expression/replacement/flags   (2)
X   Substitute the replacement for instances of the regular expression
Xin the current text buffer. Any character may be used instead of `/'.
XFor a fuller description see ed (1).
X   Flags is zero or more of the following:
X
Xg -- Global. Substitute for all nonoverlapping instances of the string
X   rather than just thefirst one.
X
Xp -- Print the pattern space if a replacement was made.
X
Xw -- Write. Append the current text buffer to a file argument as in a
X   w command if a replacement is made. Standard output is used if no
X   file argument is given
X
Xt "label" (2)
X   Branch-if-test. Branch to the : command with the given label if any
Xsubstitutes have been made since the most recent read of an input line
Xor execution of a `t'or `T'. If no label is given, branch to the end
Xof the script.
X
XT "label" (2)
X   Branch-on-error. Branch to the : command with the given label if no
Xsubstitutes have succeeded since the last input line or t or T command.
XBranch to the end of the script if no label is given.
X
Xw "wfile" (2)
X   Write. Append the current text buffer to wfile .
X
XW "wfile" (2)
X   Write first line. Append first line of the current text buffer
Xto wfile.
X

Xx   (2)
X   Exchange the contents of the current text buffer and hold space.
X
Xy /string1/string2/   (2)
X   Translate. Replace each occurrence of a character in string1 with
Xthe corresponding character in string2. The lengths of these strings
Xmust be equal.
X
X! "command"   (2)
X   All-but. Apply the function (or group, if function is `{') only to
Xlines not selected by the address(es).
X
X: "label" (0)
X   This command does nothing but hold a label for `b' and `t' commands
Xto branch to.
X
X=   (1)
X   Place the current line number on the standard output as a line.
X
X{   (2)
X   Execute the following commands through a matching `}' only when the
Xcurrent line matches the address or address range given.
X
X   An empty command is ignored.
X
X
XPORTABILITY
X   This tool was reverse-engineered from BSD 4.1 UNIX sed, and (as far
Xas the author's knowledge and tests can determine) is compatible with
Xit. All documented features of BSD 4.1 sed are supported.
X
X   One undocumented feature (a leading 'n' in the first comment having
Xthe same effect as an -n command-line option) has been omitted.
X
XThe following bugs and limitations have been fixed:
X   * There is no hidden length limit (40 in BSD sed) on w file names.
X   * There is no limit (8 in BSD sed) on the length of labels.
X   * The exchange command now works for long pattern and hold spaces.
X
XThe following enhancements to existing commands have been made:
X   * a, i commands don't insist on a leading backslash-\n in the text.
X   * r, w commands don't insist on whitespace before the filename.
X   * The g, p and P options on s commands may be given in any order.
X
XSome enhancements to regular-expression syntax have been made:
X   * \t is recognized in REs (and elswhere) as an escape for tab.
X   * In an RE, + calls for 1..n repeats of the previous pattern.
X
XThe following are completely new features:
X   * The l command (list, undocumented and weaker in BSD)
X   * The W command (write first line of pattern space to file).
X   * The T command (branch on last substitute failed).
X   * Trailing comments are now allowed on command lines.

```
X
X   In addition,  sed's error messages have been made more specific and
Xinformative.
X
X   The implementation is also  significantly  smaller and  faster than
XBSD 4.1 sed. It uses only the standard I/O library and exit.
X
X
XNOTE
X   This is a freeware component of the GNU  operating system. The user
Xis hereby granted permission to use, modify, reproduce and  distribute
Xit subject to the following conditions:
X   1. The authorship notice appearing  in  each source file may not be
Xaltered or deleted.
X   2. The object form may not be distributed without source.
X
X
XSEE ALSO
X   ed(1), grep(1), awk(1), lex(1), regexp(5)
END_OF_sed.man
if test 9360 -ne `wc -c <sed.man`; then
    echo shar: \"sed.man\" unpacked with wrong size!
fi
# end of overwriting check
fi
if test -f sedcomp.c -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"sedcomp.c\"
else
echo shar: Extracting \"sedcomp.c\" \(37223 characters\)
sed "s/^X//" >sedcomp.c <<'END_OF_sedcomp.c'
X#if 0                  /* SAJ */
X#include "compiler.h"
X#ifdef LATTICE
X#define void int
X#endif
X#endif                 /* SAJ */
X
X#include "debug.h"
X/* sedcomp.c -- stream editor main and compilation phase
X
X   The stream editor compiles its command input  (from files or -e options)
Xinto an internal form using compile() then executes the compiled form using
Xexecute(). Main() just initializes data structures, interprets command line
Xoptions, and calls compile() and execute() in appropriate sequence.
X   The data structure produced by compile() is an array of compiled-command
Xstructures (type sedcmd).  These contain several pointers into pool[], the
Xregular-expression and text-data pool, plus a command code and g & p flags.
XIn the special case that the command is a label the struct  will hold a ptr
Xinto the labels array labels[] during most of the compile,  until resolve()
Xresolves references at the end.
X   The operation of execute() is described in its source module.
X
X==== Written for the GNU operating system by Eric S. Raymond ==== */
```

```
X
X#include <stdio.h>              /* uses getc, fprintf, fopen, fclose */
X#include "sed.h"                /* command type struct and name defines */
X
X/* imported functions */
Xextern int strcmp();            /* test strings for equality */
Xextern void execute();          /* execute compiled command */
X
X/***** public stuff ******/
X
X#define MAXCMDS         200     /* maximum number of compiled commands */
X#define MAXLINES        256     /* max # numeric addresses to compile */
X
X/* main data areas */
Xchar    linebuf[MAXBUF+1];      /* current-line buffer */
Xsedcmd  cmds[MAXCMDS+1];        /* hold compiled commands */
Xlong    linenum[MAXLINES];      /* numeric-addresses table */
X
X/* miscellaneous shared variables */
Xint     nflag;                  /* -n option flag */
Xint     eargc;                  /* scratch copy of argument count */
Xsedcmd  *pending        = NULL; /* next command to be executed */
Xchar    bits[]          = {1,2,4,8,16,32,64,128};
X
X/***** module common stuff *****/
X
X#define POOLSIZE        10000   /* size of string-pool space */
X#define WFILES          10      /* max # w output files that can be compiled */
X#define RELIMIT         256     /* max chars in compiled RE */
X#define MAXDEPTH        20      /* maximum {}-nesting level */
X#define MAXLABS         50      /* max # of labels that can be handled */
X
X#define SKIPWS(pc)      while ((*pc==' ') || (*pc=='\t')) pc++
X#define ABORT(msg)      (fprintf(stderr, msg, linebuf), exit(2))
X#define IFEQ(x, v)      if (*x == v) x++ , /* do expression */
X
X
/* error messages */
Xstatic char     AGMSG[] = "sed: garbled address %s\n";
Xstatic char     CGMSG[] = "sed: garbled command %s\n";
Xstatic char     TMTXT[] = "sed: too much text: %s\n";
Xstatic char     AD1NG[] = "sed: no addresses allowed for %s\n";
Xstatic char     AD2NG[] = "sed: only one address allowed for %s\n";
Xstatic char     TMCDS[] = "sed: too many commands, last was %s\n";
Xstatic char     COCFI[] = "sed: cannot open command-file %s\n";
Xstatic char     UFLAG[] = "sed: unknown flag %c\n";
Xstatic char     COOFI[] = "sed: cannot open %s\n";
Xstatic char     CCOFI[] = "sed: cannot create %s\n";
Xstatic char     ULABL[] = "sed: undefined label %s\n";
Xstatic char     TMLBR[] = "sed: too many {'s\n";
Xstatic char     FRENL[] = "sed: first RE must be non-null\n";
Xstatic char     NSCAX[] = "sed: no such command as %s\n";
Xstatic char     TMRBR[] = "sed: too many }'s\n";
```

```
Xstatic char     DLABL[] = "sed: duplicate label %s\n";
Xstatic char     TMLAB[] = "sed: too many labels: %s\n";
Xstatic char     TMWFI[] = "sed: too many w files\n";
Xstatic char     REITL[] = "sed: RE too long: %s\n";
Xstatic char     TMLNR[] = "sed: too many line numbers\n";
Xstatic char     TRAIL[] = "sed: command \"%s\" has trailing garbage\n";
X
Xtypedef struct                 /* represent a command label */
X{
X        char           *name;          /* the label name */
X        sedcmd         *last;          /* it's on the label search list */
X        sedcmd         *address;       /* pointer to the cmd it labels */
X}
Xlabel;
X
X/* label handling */
Xstatic label    labels[MAXLABS];        /* here's the label table */
Xstatic label    *lab    = labels + 1;   /* pointer to current label */
Xstatic label    *lablst = labels;       /* header for search list */
X
X/* string pool for regular expressions, append text, etc. etc. */
Xstatic char     pool[POOLSIZE];                 /* the pool */
Xstatic char     *fp     = pool;                 /* current pool pointer */
Xstatic char     *poolend = pool + POOLSIZE;     /* pointer past pool end */
X
X/* compilation state */
Xstatic FILE     *cmdf   = NULL;         /* current command source */
Xstatic char     *cp     = linebuf;      /* compile pointer */
Xstatic sedcmd   *cmdp   = cmds;         /* current compiled-cmd ptr */
Xstatic char     *lastre = NULL;         /* old RE pointer */
Xstatic int      bdepth  = 0;            /* current {}-nesting level */
Xstatic int      bcount  = 0;            /* # tagged patterns in current RE */
Xstatic char     **eargv;                /* scratch copy of argument list */
X
X/* compilation flags */
Xstatic int      eflag;                  /* -e option flag */
Xstatic int      gflag;                  /* -g option flag */
X
X

Xmain(argc, argv)
X/* main sequence of the stream editor */
Xint     argc;
Xchar    *argv[];
X{
X        void compile(), resolve();
X
X        eargc   = argc;         /* set local copy of argument count */
X        eargv   = argv;         /* set local copy of argument list */
X        cmdp->addr1 = pool;     /* 1st addr expand will be at pool start */
X        if (eargc == 1)
X                exit(0);        /* exit immediately if no arguments */
XPASS("main(): setup");
```

```
X	/* scan through the arguments, interpreting each one */
X	while ((--eargc > 0) && (**++eargv == '-'))
X		switch (eargv[0][1])
X		{
X		case 'e':
X			eflag++; compile();	/* compile with e flag on */
X			eflag = 0;
X			continue;		/* get another argument */
X		case 'f':
X			if (eargc-- <= 0)	/* barf if no -f file */
X				exit(2);
X			if ((cmdf = fopen(*++eargv, "r")) == NULL)
X			{
X				fprintf(stderr, COCFI, *eargv);
X				exit(2);
X			}
X			compile();	/* file is O.K., compile it */
X			fclose(cmdf);
X			continue;	/* go back for another argument */
X		case 'g':
X			gflag++;	/* set global flag on all s cmds */
X			continue;
X		case 'n':
X			nflag++;	/* no print except on p flag or w */
X			continue;
X		default:
X			fprintf(stdout, UFLAG, eargv[0][1]);
X			continue;
X		}
X
XPASS("main(): argscan");
X
X	if (cmdp == cmds)	/* no commands have been compiled */
X	{
X		eargv--; eargc++;
X		eflag++; compile(); eflag = 0;
X		eargv++; eargc--;
X	}
X
X	if (bdepth)	/* we have unbalanced squigglies */
X		ABORT(TMLBR);
X
X	lablst->address = cmdp; /* set up header of label linked list */
X	resolve();		/* resolve label table indirections */
XPASS("main(): resolve");
X	if (eargc <= 0)		/* if there were no -e commands */
X		execute(NULL);	/*   execute commands from stdin only */
X	else while(--eargc>=0)	/* else execute only -e commands */
X		execute(*eargv++);
XPASS("main(): end & exit OK");
X	exit(0);		/* everything was O.K. if we got here */
X}
X
```

```
X

X#define H       0x80    /* 128 bit, on if there's really code for command */
X#define LOWCMD  56      /* = '8', lowest char indexed in cmdmask */
X
X/* indirect through this to get command internal code, if it exists */
Xstatic char     cmdmask[] =
X{
X        0,      0,      H,      0,      0,      H+EQCMD,0,      0,
X        0,      0,      0,      0,      H+CDCMD,0,      0,      CGCMD,
X        CHCMD,  0,      0,      0,      0,      0,      CNCMD,  0,
X        CPCMD,  0,      0,      0,      H+CTCMD,0,      0,      H+CWCMD,
X        0,      0,      0,      0,      0,      0,      0,      0,
X        0,      H+ACMD, H+BCMD, H+CCMD, DCMD,   0,      0,      GCMD,
X        HCMD,   H+ICMD, 0,      0,      H+LCMD, 0,      NCMD,   0,
X        PCMD,   H+QCMD, H+RCMD, H+SCMD, H+TCMD, 0,      0,      H+WCMD,
X        XCMD,   H+YCMD, 0,      H+BCMD, 0,      H,      0,      0,
X};
X
Xstatic void compile()
X/* precompile sed commands out of a file */
X{
X        char            ccode, *address();
X
XPASS("compile(): entry");
X
X        for(;;)                                 /* main compilation loop */
X        {
X                if (*cp != ';')                 /* get a new command line */
X                        if (cmdline(cp = linebuf) < 0)
X                                break;
X                SKIPWS(cp);
X                if (*cp=='\0' || *cp=='#')      /* a comment */
X                        continue;
X                if (*cp == ';')                 /* ; separates cmds */
X                {
X                        cp++;
X                        continue;
X                }
X
X                /* compile first address */
X                if (fp > poolend)
X                        ABORT(TMTXT);
X                else if ((fp = address(cmdp->addr1 = fp)) == BAD)
X                        ABORT(AGMSG);
X
X                if (fp == cmdp->addr1)          /* if empty RE was found */
X                {
X                        if (lastre)             /* if there was previous RE */
X                                cmdp->addr1 = lastre;   /* use it */
X                        else
X                                ABORT(FRENL);
X                }
```

```
X                else if (fp == NULL)            /* if fp was NULL */
X                {
X                        fp = cmdp->addr1;       /* use current pool location */
X                        cmdp->addr1 = NULL;
X                }
X                else
X                {
X                        lastre = cmdp->addr1;
X                        if (*cp == ',' || *cp == ';')   /* there's 2nd addr */
X                        {
X                                cp++;
X                                if (fp > poolend) ABORT(TMTXT);
X                                fp = address(cmdp->addr2 = fp);
X                                if (fp == BAD || fp == NULL) ABORT(AGMSG);
X                                if (fp == cmdp->addr2)
X                                        cmdp->addr2 = lastre;
X                                else
X                                        lastre = cmdp->addr2;
X                        }
X                        else
X                                cmdp->addr2 = NULL;     /* no 2nd address */
X                }
X                if (fp > poolend) ABORT(TMTXT);
X
X                SKIPWS(cp);             /* discard whitespace after address */
X                IFEQ(cp, '!') cmdp->flags.allbut = 1;
X
X                SKIPWS(cp);             /* get cmd char, range-check it */
X                if ((*cp < LOWCMD) || (*cp > '~')
X                        || ((ccode = cmdmask[*cp - LOWCMD]) == 0))
X                                ABORT(NSCAX);
X
X                cmdp->command = ccode & ~H;     /* fill in command value */
X                if ((ccode & H) == 0)           /* if no compile-time code */
X                        cp++;                   /* discard command char */
X                else if (cmdcomp(*cp++))        /* execute it; if ret = 1 */
X                        continue;               /* skip next line read */
X
X                if (++cmdp >= cmds + MAXCMDS) ABORT(TMCDS);
X
X                SKIPWS(cp);                     /* look for trailing stuff */
X                if (*cp != '\0')
X                        if (*++cp == ';')
X                                continue;
X                        else if (cp[-1] != '#')
X                                ABORT(TRAIL);
X        }
X}
X
Xstatic int cmdcomp(cchar)
X/* compile a single command */
Xregister char   cchar;          /* character name of command */
X{
```

```
X       char            *gettext(), *rhscomp(), *recomp(), *ycomp();
X       static sedcmd   **cmpstk[MAXDEPTH];     /* current cmd stack for {} */
X       static char     *fname[WFILES];         /* w file name pointers */
X       static FILE     *fout[WFILES]={stdout}; /* w file file ptrs */
X       static int      nwfiles = 1;            /* count of open w files */
X       int             i;                      /* indexing dummy used in w */
X       sedcmd          *sp1, *sp2;             /* temps for label searches */
X       label           *lpt, *search();        /* ditto, and the searcher */
X       char            redelim;                /* current RE delimiter */
X
X       switch(cchar)
X       {
X       case '{':       /* start command group */
X               cmdp->flags.allbut = !cmdp->flags.allbut;
X               cmpstk[bdepth++] = &(cmdp->u.link);
X               if (++cmdp >= cmds + MAXCMDS) ABORT(TMCDS);
X               if (*cp == '\0') *cp = ';';     /* get next cmd w/o lineread */
X               return(1);
X
X       case '}':       /* end command group */
X               if (cmdp->addr1) ABORT(AD1NG);  /* no addresses allowed */
X               if (--bdepth < 0) ABORT(TMRBR); /* too many right braces */
X               *cmpstk[bdepth] = cmdp;         /* set the jump address */
X               return(1);
X
X       case '=':                       /* print current source line number */
X       case 'q':                       /* exit the stream editor */
X               if (cmdp->addr2) ABORT(AD2NG);
X               break;
X
X       case ':':       /* label declaration */
X               if (cmdp->addr1) ABORT(AD1NG);  /* no addresses allowed */
X               fp = gettext(lab->name = fp);   /* get the label name */
X               if (lpt = search(lab))          /* does it have a double? */
X               {
X                       if (lpt->address) ABORT(DLABL); /* yes, abort */
X               }
X               else    /* check that it doesn't overflow label table */
X               {
X                       lab->last = NULL;
X                       lpt = lab;
X                       if (++lab >= labels + MAXLABS) ABORT(TMLAB);
X               }
X               lpt->address = cmdp;
X               return(1);
X
X       case 'b':       /* branch command */
X       case 't':       /* branch-on-succeed command */
X       case 'T':       /* branch-on-fail command */
X               SKIPWS(cp);
X               if (*cp == '\0')        /* if branch is to start of cmds... */
X               {
X                       /* add current command to end of label last */
```

```
X                       if (sp1 = lablst->last)
X                       {
X                               while(sp2 = sp1->u.link)
X                                       sp1 = sp2;
X                               sp1->u.link = cmdp;
X                       }
X                       else    /* lablst->last == NULL */
X                               lablst->last = cmdp;
X                       break;
X               }
X               fp = gettext(lab->name = fp);   /* else get label into pool */
X               if (lpt = search(lab))          /* enter branch to it */
X               {
X                       if (lpt->address)
X                               cmdp->u.link = lpt->address;
X                       else
X                       {
X                               sp1 = lpt->last;
X                               while(sp2 = sp1->u.link)
X                                       sp1 = sp2;
X                               sp1->u.link = cmdp;
X                       }
X               }
X               else            /* matching named label not found */
X               {
X                       lab->last = cmdp;       /* add the new label */
X                       lab->address = NULL;    /* it's forward of here */
X                       if (++lab >= labels + MAXLABS)  /* overflow if last */
X                               ABORT(TMLAB);
X               }
X               break;
X
X       case 'a':       /* append text */
X       case 'i':       /* insert text */
X       case 'r':       /* read file into stream */
X               if (cmdp->addr2) ABORT(AD2NG);
X       case 'c':       /* change text */
X               if ((*cp == '\\') && (*++cp == '\n')) cp++;
X               fp = gettext(cmdp->u.lhs = fp);
X               break;
X
X       case 'D':       /* delete current line in hold space */
X               cmdp->u.link = cmds;
X               break;
X
X       case 's':       /* substitute regular expression */
X               redelim = *cp++;                /* get delimiter from 1st ch */
X               if ((fp = recomp(cmdp->u.lhs = fp, redelim)) == BAD)
X                       ABORT(CGMSG);
X               if (fp == cmdp->u.lhs)          /* if compiled RE zero len */
X                       cmdp->u.lhs = lastre;   /*   use the previous one */
X               else                            /* otherwise */
X                       lastre = cmdp->u.lhs;   /*   save the one just found */
```

```
X               if ((cmdp->rhs = fp) > poolend) ABORT(TMTXT);
X               if ((fp = rhscomp(cmdp->rhs, redelim)) == BAD) ABORT(CGMSG);
X               if (gflag) cmdp->flags.global++;
X               while (*cp == 'g' || *cp == 'p' || *cp == 'P')
X               {
X                       IFEQ(cp, 'g') cmdp->flags.global++;
X                       IFEQ(cp, 'p') cmdp->flags.print = 1;
X                       IFEQ(cp, 'P') cmdp->flags.print = 2;
X               }
X
X       case 'l':       /* list pattern space */
X               if (*cp == 'w')
X                       cp++;           /* and execute a w command! */
X               else
X                       break;          /* s or l is done */
X
X       case 'w':       /* write-pattern-space command */
X       case 'W':       /* write-first-line command */
X               if (nwfiles >= WFILES) ABORT(TMWFI);
X               fp=gettext(fname[nwfiles]=fp);  /* filename will be in pool */
X               for(i = nwfiles-1; i >= 0; i--) /* match it in table */
X                       if (strcmp(fname[nwfiles], fname[i]) == 0)
X                       {
X                               cmdp->fout = fout[i];
X                               return(0);
X                       }
X               /* if didn't find one, open new out file */
X               if ((cmdp->fout = fopen(fname[nwfiles], "w")) == NULL)
X               {
X                       fprintf(stderr, CCOFI, fname[nwfiles]);
X                       exit(2);
X               }
X               fout[nwfiles++] = cmdp->fout;
X               break;
X
X       case 'y':       /* transliterate text */
X               fp = ycomp(cmdp->u.lhs = fp, *cp++);    /* compile translit */
X               if (fp == BAD) ABORT(CGMSG);            /* fail on bad form */
X               if (fp > poolend) ABORT(TMTXT);         /* fail on overflow */
X               break;
X       }
X       return(0);      /* succeeded in interpreting one command */
X}
X
Xstatic char *rhscomp(rhsp, delim)       /* uses bcount */
X/* generate replacement string for substitute command right hand side */
Xregister char   *rhsp;          /* place to compile expression to */
Xregister char   delim;          /* regular-expression end-mark to look for */
X{
X       register char   *p = cp;                /* strictly for speed */
X
X       for(;;)
X               if ((*rhsp = *p++) == '\\')     /* copy; if it's a \, */
```

```
X               {
X                       *rhsp = *p++;           /* copy escaped char */
X                       /* check validity of pattern tag */
X                       if (*rhsp > bcount + '0' && *rhsp <= '9')
X                               return(BAD);
X                       *rhsp++ |= 0x80;        /* mark the good ones */
X                       continue;
X               }
X               else if (*rhsp == delim)        /* found RE end, hooray... */
X               {
X                       *rhsp++ = '\0';         /* cap the expression string */
X                       cp = p;
X                       return(rhsp);           /* pt at 1 past the RE */
X               }
X               else if (*rhsp++ == '\0')       /* last ch not RE end, help! */
X                       return(BAD);
X}
X
Xstatic char *recomp(expbuf, redelim)    /* uses cp, bcount */
X/* compile a regular expression to internal form */
Xchar    *expbuf;                        /* place to compile it to */
Xchar    redelim;                        /* RE end-marker to look for */
X{
X       register char   *ep = expbuf;   /* current-compiled-char pointer */
X       register char   *sp = cp;       /* source-character ptr */
X       register int    c;              /* current-character pointer */
X       char            negclass;       /* all-but flag */
X       char            *lastep;        /* ptr to last expr compiled */
X       char            *svclass;       /* start of current char class */
X       char            brnest[MAXTAGS];        /* bracket-nesting array */
X       char            *brnestp;       /* ptr to current bracket-nest */
X       char            *pp;            /* scratch pointer */
X       int             classct;        /* class element count */
X       int             tags;           /* # of closed tags */
X
X       if (*cp == redelim)             /* if first char is RE endmarker */
X               return(cp++, expbuf);   /* leave existing RE unchanged */
X
X       lastep = NULL;                  /* there's no previous RE */
X       brnestp = brnest;               /* initialize ptr to brnest array */
X       tags = bcount = 0;              /* initialize counters */
X
X       if (*ep++ = (*sp == '^'))       /* check for start-of-line syntax */
X               sp++;
X
X       for (;;)
X       {
X               if (ep >= expbuf + RELIMIT)     /* match is too large */
X                       return(cp = sp, BAD);
X               if ((c = *sp++) == redelim)     /* found the end of the RE */
X               {
X                       cp = sp;
X                       if (brnestp != brnest)  /* \(, \) unbalanced */
```

```
X                       return(BAD);
X               *ep++ = CEOF;           /* write end-of-pattern mark */
X               return(ep);             /* return ptr to compiled RE */
X       }
X       if ((c != '*') && (c != '+'))   /* if we're a postfix op */
X               lastep = ep;            /*   get ready to match last */
X
X       switch (c)
X       {
X       case '\\':
X               if ((c = *sp++) == '(') /* start tagged section */
X               {
X                       if (bcount >= MAXTAGS)
X                               return(cp = sp, BAD);
X                       *brnestp++ = bcount;    /* update tag stack */
X                       *ep++ = CBRA;           /* enter tag-start */
X                       *ep++ = bcount++;       /* bump tag count */
X                       continue;
X               }
X               else if (c == ')')      /* end tagged section */
X               {
X                       if (brnestp <= brnest)  /* extra \) */
X                               return(cp = sp, BAD);
X                       *ep++ = CKET;           /* enter end-of-tag */
X                       *ep++ = *--brnestp;     /* pop tag stack */
X                       tags++;                 /* count closed tags */
X                       continue;
X               }
X               else if (c >= '1' && c <= '9')  /* tag use */
X               {
X                       if ((c -= '1') >= tags) /* too few */
X                               return(BAD);
X                       *ep++ = CBACK;          /* enter tag mark */
X                       *ep++ = c;              /* and the number */
X                       continue;
X               }
X               else if (c == '\n')     /* escaped newline no good */
X                       return(cp = sp, BAD);
X               else if (c == 'n')              /* match a newline */
X                       c = '\n';
X               else if (c == 't')              /* match a tab */
X                       c = '\t';
X               else
X                       goto defchar;           /* else match \c */
X
X       case '\0':      /* ignore nuls */
X               continue;
X
X       case '\n':      /* trailing pattern delimiter is missing */
X               return(cp = sp, BAD);
X
X       case '.':       /* match any char except newline */
X               *ep++ = CDOT;
```

```
X                       continue;
X
X               case '+':       /* 1 to n repeats of previous pattern */
X                       if (lastep == NULL)     /* if + not first on line */
X                               goto defchar;   /*   match a literal + */
X                       if (*lastep == CKET)    /* can't iterate a tag */
X                               return(cp = sp, BAD);
X                       pp = ep;                /* else save old ep */
X                       while (lastep < pp)     /* so we can blt the pattern */
X                               *ep++ = *lastep++;
X                       *lastep |= STAR;        /* flag the copy */
X                       continue;
X
X               case '*':       /* 0..n repeats of previous pattern */
X                       if (lastep == NULL)     /* if * isn't first on line */
X                               goto defchar;   /*   match a literal * */
X                       if (*lastep == CKET)    /* can't iterate a tag */
X                               return(cp = sp, BAD);
X                       *lastep |= STAR;        /* flag previous pattern */
X                       continue;
X
X               case '$':       /* match only end-of-line */
X                       if (*sp != redelim)     /* if we're not at end of RE */
X                               goto defchar;   /*   match a literal $ */
X                       *ep++ = CDOL;           /* insert end-symbol mark */
X                       continue;
X
X               case '[':       /* begin character set pattern */
X                       if (ep + 17 >= expbuf + RELIMIT)
X                               ABORT(REITL);
X                       *ep++ = CCL;            /* insert class mark */
X                       if (negclass = ((c = *sp++) == '^'))
X                               c = *sp++;
X                       svclass = sp;           /* save ptr to class start */
X                       do {
X                               if (c == '\0') ABORT(CGMSG);
X
X                               /* handle character ranges */
X                               if (c == '-' && sp > svclass && *sp != ']')
X                                       for (c = sp[-2]; c < *sp; c++)
X                                               ep[c >> 3] |= bits[c & 7];
X
X                               /* handle escape sequences in sets */
X                               if (c == '\\')
X                                       if ((c = *sp++) == 'n')
X                                               c = '\n';
X                                       else if (c == 't')
X                                               c = '\t';
X
X                               /* enter (possibly translated) char in set */
X                               ep[c >> 3] |= bits[c & 7];
X                       } while
X                               ((c = *sp++) != ']');
```

```
X
X                       /* invert the bitmask if all-but was specified */
X                       if (negclass)
X                               for(classct = 0; classct < 16; classct++)
X                                       ep[classct] ^= 0xFF;
X                       ep[0] &= 0xFE;          /* never match ASCII 0 */
X                       ep += 16;               /* advance ep past set mask */
X                       continue;
X
X               defchar:        /* match literal character */
X               default:        /* which is what we'd do by default */
X                       *ep++ = CCHR;           /* insert character mark */
X                       *ep++ = c;
X               }
X       }
X}
X
Xstatic int cmdline(cbuf)                /* uses eflag, eargc, cmdf */
X/* read next command from -e argument or command file */
Xregister char   *cbuf;
X{
X       register int    inc;    /* not char because must hold EOF */
X
X       cbuf--;                 /* so pre-increment points us at cbuf */
X
X       /* e command flag is on */
X       if (eflag)
X       {
X               register char   *p;     /* ptr to current -e argument */
X               static char     *savep; /* saves previous value of p */
X
X               if (eflag > 0)  /* there are pending -e arguments */
X               {
X                       eflag = -1;
X                       if (eargc-- <= 0)
X                               exit(2);        /* if no arguments, barf */
X
X                       /* else transcribe next e argument into cbuf */
X                       p = *++eargv;
X                       while(*++cbuf = *p++)
X                               if (*cbuf == '\\')
X                               {
X                                       if ((*++cbuf = *p++) == '\0')
X                                               return(savep = NULL, -1);
X                                       else
X                                               continue;
X                               }
X                               else if (*cbuf == '\n') /* end of 1 cmd line */
X                               {
X                                       *cbuf = '\0';
X                                       return(savep = p, 1);
X                                       /* we'll be back for the rest... */
X                               }
```

```
X
X                       /* found end-of-string; can advance to next argument */
X                       return(savep = NULL, 1);
X               }
X
X               if ((p = savep) == NULL)
X                       return(-1);
X
X               while(*++cbuf = *p++)
X                       if (*cbuf == '\\')
X                       {
X                               if ((*++cbuf = *p++) == '0')
X                                       return(savep = NULL, -1);
X                               else
X                                       continue;
X                       }
X                       else if (*cbuf == '\n')
X                       {
X                               *cbuf = '\0';
X                               return(savep = p, 1);
X                       }
X
X               return(savep = NULL, 1);
X       }
X
X       /* if no -e flag read from command file descriptor */
X       while((inc = getc(cmdf)) != EOF)                /* get next char */
X               if ((*++cbuf = inc) == '\\')            /* if it's escape */
X                       *++cbuf = inc = getc(cmdf);     /* get next char */
X               else if (*cbuf == '\n')                 /* end on newline */
X                       return(*cbuf = '\0', 1);        /* cap the string */
X
X       return(*++cbuf = '\0', -1);     /* end-of-file, no more chars */
X}
X
Xstatic char *address(expbuf)            /* uses cp, linenum */
X/* expand an address at *cp... into expbuf, return ptr at following char */
Xregister char   *expbuf;
X{
X       static int      numl = 0;       /* current ind in addr-number table */
X       register char   *rcp;           /* temp compile ptr for forwd look */
X       long            lno;            /* computed value of numeric address */
X
X       if (*cp == '$')                 /* end-of-source address */
X       {
X               *expbuf++ = CEND;       /* write symbolic end address */
X               *expbuf++ = CEOF;       /* and the end-of-address mark (!) */
X               cp++;                   /* go to next source character */
X               return(expbuf);         /* we're done */
X       }
X       if (*cp == '/')                 /* start of regular-expression match */
X               return(recomp(expbuf, *cp++));  /* compile the RE */
X
```

```
X	rcp = cp; lno = 0;		/* now handle a numeric address */
X	while(*rcp >= '0' && *rcp <= '9')	/* collect digits */
X		lno = lno*10 + *rcp++ - '0';	/*  compute their value */
X
X	if (rcp > cp)			/* if we caught a number... */
X	{
X		*expbuf++ = CLNUM;	/* put a numeric-address marker */
X		*expbuf++ = numl;	/* and the address table index */
X		linenum[numl++] = lno;	/* and set the table entry */
X		if (numl >= MAXLINES)	/* oh-oh, address table overflow */
X			ABORT(TMLNR);	/*   abort with error message */
X		*expbuf++ = CEOF;	/* write the end-of-address marker */
X		cp = rcp;		/* point compile past the address */
X		return(expbuf);		/* we're done */
X	}
X
X	return(NULL);		/* no legal address was found */
X}
X
Xstatic char *gettext(txp)		/* uses global cp */
X/* accept multiline input from *cp..., discarding leading whitespace */
Xregister char   *txp;			/* where to put the text */
X{
X	register char   *p = cp;	/* this is for speed */
X
X	SKIPWS(p);			/* discard whitespace */
X	do {
X		if ((*txp = *p++) == '\\')	/* handle escapes */
X			*txp = *p++;
X		if (*txp == '\0')		/* we're at end of input */
X			return(cp = --p, ++txp);
X		else if (*txp == '\n')		/* also SKIPWS after newline */
X			SKIPWS(p);
X	} while
X		(txp++);		/* keep going till we find that nul */
X}
X
Xstatic label *search(ptr)			/* uses global lablst */
X/* find the label matching *ptr, return NULL if none */
Xregister label  *ptr;
X{
X	register label  *rp;
X	for(rp = lablst; rp < ptr; rp++)
X		if (strcmp(rp->name, ptr->name) == 0)
X			return(rp);
X	return(NULL);
X}
X
Xstatic void resolve()			/* uses global lablst */
X/* write label links into the compiled-command space */
X{
X	register label		*lptr;
X	register sedcmd		*rptr, *trptr;
```

```
X
X       /* loop through the label table */
X       for(lptr = lablst; lptr < lab; lptr++)
X               if (lptr->address == NULL)      /* barf if not defined */
X               {
X                       fprintf(stderr, ULABL, lptr->name);
X                       exit(2);
X               }
X               else if (lptr->last)            /* if last is non-null */
X               {
X                       rptr = lptr->last;              /* chase it */
X                       while(trptr = rptr->u.link)     /* resolve refs */
X                       {
X                               rptr->u.link = lptr->address;
X                               rptr = trptr;
X                       }
X                       rptr->u.link = lptr->address;
X               }
X}
X
Xstatic char *ycomp(ep, delim)
X/* compile a y (transliterate) command */
Xregister char   *ep;            /* where to compile to */
Xchar            delim;          /* end delimiter to look for */
X{
X       register char   c, *tp, *sp;
X
X       /* scan the 'from' section for invalid chars */
X       for(sp = tp = cp; *tp != delim; tp++)
X       {
X               if (*tp == '\\')
X                       tp++;
X               if ((*tp == '\n') || (*tp == '\0'))
X                       return(BAD);
X       }
X       tp++;           /* tp now points at first char of 'to' section */
X
X       /* now rescan the 'from' section */
X       while((c = *sp++ & 0x7F) != delim)
X       {
X               if (c == '\\' && *sp == 'n')
X               {
X                       sp++;
X                       c = '\n';
X               }
X               if ((ep[c] = *tp++) == '\\' && *tp == 'n')
X               {
X                       ep[c] = '\n';
X                       tp++;
X               }
X               if ((ep[c] == delim) || (ep[c] == '\0'))
X                       return(BAD);
X       }
```

```
X
X       if (*tp != delim)       /* 'to', 'from' parts have unequal lengths */
X               return(BAD);
X
X       cp = ++tp;                      /* point compile ptr past translit */
X
X       for(c = 0; c < 128; c++)        /* fill in self-map entries in table */
X               if (ep[c] == 0)
X                       ep[c] = c;
X
X       return(ep + 0x80);      /* return first free location past table end */
X}
X
X/* sedcomp.c ends here */
END_OF_sedcomp.c
if test 37223 -ne `wc -c <sedcomp.c`; then
    echo shar: \"sedcomp.c\" unpacked with wrong size!
fi
# end of overwriting check
fi
echo shar: End of shell archive.
exit 0


From 7011689391004732414
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,5d873e72a43cd21c,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-02-24 09:37:30 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i016: sed -- GNU sed (stream editor)  part02/02
Keywords: shar
Message-ID: <6462@saturn.ucsc.edu>
Date: 24 Feb 89 17:37:30 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 802
Approved: koreth@ssyx.ucsc.edu

Submitted-by: saj@chinet.chi.il.us (Stephen Jacobs)
Posting-number: Volume 2, Issue 16
Archive-name: sed/part02

#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of shell archive."
# Contents:  sedexec.c sedtest.bat
```

```
# Wrapped by saj@chinet on Mon Jan 30 20:12:03 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f sedexec.c -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"sedexec.c\"
else
echo shar: Extracting \"sedexec.c\" \(28811 characters\)
sed "s/^X//" >sedexec.c <<'END_OF_sedexec.c'
X#if 0                      /* SAJ */
X#include "compiler.h"
X#ifdef LATTICE
X#define void int
X#endif
X
X#endif                      /* SAJ */
X#include "debug.h"
X/*
Xsedexec.c -- execute compiled form of stream editor commands
X
X   The single entry point of this module is the function execute(). It
Xmay take a string argument (the name of a file to be used as text)  or
Xthe argument NULL which tells it to filter standard input. It executes
Xthe compiled commands in cmds[] on each line in turn.
X   The function command() does most of the work. Match() and advance()
Xare used for matching text against precompiled regular expressions and
Xdosub() does right-hand-side substitution.  Getline() does text input;
Xreadout() and memcmp() are output and string-comparison utilities.
X
X==== Written for the GNU operating system by Eric S. Raymond ====
X*/
X
X#include <stdio.h>      /* {f}puts, {f}printf, getc/putc, f{re}open, fclose */
X#include <ctype.h>      /* for isprint(), isdigit(), toascii() macros */
X#include "sed.h"        /* command type structures & miscellaneous constants */
X
Xextern char     *strcpy();      /* used in dosub */
X
X/***** shared variables imported from the main ******/
X
X/* main data areas */
Xextern char     linebuf[];      /* current-line buffer */
Xextern sedcmd   cmds[];         /* hold compiled commands */
Xextern long     linenum[];      /* numeric-addresses table */
X
X/* miscellaneous shared variables */
Xextern int      nflag;          /* -n option flag */
Xextern int      eargc;          /* scratch copy of argument count */
Xextern sedcmd   *pending;       /* ptr to command waiting to be executed */
Xextern char     bits[];         /* the bits table */
X
X/***** end of imported stuff *****/
X
X#define MAXHOLD         MAXBUF  /* size of the hold space */
X#if 0        /* I had to leave this in.  Is it a joke, or what?
```

```
X         *3 Hours single stepping a broken program isn't funny  SAJ */
X#define GENSIZ          71      /* maximum genbuf size */
X#endif
X#define GENSIZ          500      /* maximum genbuf size */
X#define TRUE            1
X#define FALSE           0
X
Xstatic char LTLMSG[]    = "sed: line too long\n";
X
Xstatic char     *spend;         /* current end-of-line-buffer pointer */
Xstatic long     lnum = 0L;      /* current source line number */
X
X/* append buffer maintenance */
Xstatic sedcmd   *appends[MAXAPPENDS];   /* array of ptrs to a,i,c commands */
Xstatic sedcmd   **aptr = appends;       /* ptr to current append */
X
X/* genbuf and its pointers */
Xstatic char     genbuf[GENSIZ];
Xstatic char     *lcomend = genbuf + GENSIZ;
Xstatic char     *loc1;
Xstatic char     *loc2;
Xstatic char     *locs;
X
X/* command-logic flags */
Xstatic int      lastline;               /* do-line flag */
Xstatic int      jump;                   /* jump to cmd's link address if set */
Xstatic int      delete;                 /* delete command flag */
X
X/* tagged-pattern tracking */
Xstatic char     *bracend[MAXTAGS];      /* tagged pattern start pointers */
Xstatic char     *brastart[MAXTAGS];     /* tagged pattern end pointers */
X
X

Xvoid execute(file)
X/* execute the compiled commands in cmds[] on a file */
Xchar *file;             /* name of text source file to be filtered */
X{
X        register char           *p1, *p2;       /* dummy copy ptrs */
X        register sedcmd         *ipc;           /* ptr to current command */
X        char                    *execp;         /* ptr to source */
X        char                    *getline();     /* input-getting functions */
X        void                    command(), readout();
X
XPASS("execute(): entry");
X
X        if (file != NULL)       /* filter text from a named file */
X                if (freopen(file, "r", stdin) == NULL)
X                        fprintf(stderr, "sed: can't open %s\n", file);
XPASS("execute(): reopen");
X
X        if (pending)            /* there's a command waiting */
X        {
```

```
X                ipc = pending;          /* it will be first executed */
X                pending = FALSE;        /*   turn off the waiting flag */
X                goto doit;              /*   go to execute it immediately */
X        }
X
X        /* here's the main command-execution loop */
X        for(;;)
X        {
XPASS("execute(): main loop entry");
X
X                /* get next line to filter */
X                if ((execp = getline(linebuf)) == BAD)
X                        return;
XPASS("execute(): getline");
X                spend = execp;
X
X                /* loop through compiled commands, executing them */
X                for(ipc = cmds; ipc->command; )
X                {
XPASS("execute(): command loop entry");
X                        /* all no-address commands are selected */
X                        if (ipc->addr1 && !selected(ipc))
X                        {
X                                ipc++;
X                                continue;
X                        }
X        doit:
XPASS("execute(): doit");
X                        command(ipc);   /* execute the command pointed at */
XPASS("execute(): command");
X
X                        if (delete)     /* if delete flag is set */
X                                break;  /* don't exec rest of compiled cmds */
X
X                        if (jump)       /* if jump set, follow cmd's link */
X                        {
X                                jump = FALSE;
X                                if ((ipc = ipc->u.link) == 0)
X                                {
X                                        ipc = cmds;
X                                        break;
X                                }
X                        }
X                        else            /* normal goto next command */
X                                ipc++;
XPASS("execute(): end command loop");
X                }
X                /* we've now done all modification commands on the line */
X
X                /* here's where the transformed line is output */
XPASS("execute(): output");
X                if (!nflag && !delete)
X                {
```

```
X                       for(p1 = linebuf; p1 < spend; p1++)
X                               putc(*p1, stdout);
X                       putc('\n', stdout);
X               }
X
X               /* if we've been set up for append, emit the text from it */
X               if (aptr > appends)
X                       readout();
X
X               delete = FALSE; /* clear delete flag; about to get next cmd */
XPASS("execute(): end main loop");
X       }
XPASS("execute(): end execute");
X}
X
Xstatic int selected(ipc)
X/* is current command selected */
Xsedcmd  *ipc;
X{
X       register char   *p1 = ipc->addr1;       /* point p1 at first address */
X       register char   *p2 = ipc->addr2;       /*   and p2 at second */
X       char            c;
X
X       if (ipc->flags.inrange)
X       {
X               if (*p2 == CEND)
X                       p1 = NULL;
X               else if (*p2 == CLNUM)
X               {
X                       c = p2[1];
X                       if (lnum > linenum[c])
X                       {
X                               ipc->flags.inrange = FALSE;
X                               if (ipc->flags.allbut)
X                                       return(TRUE);
X                               ipc++;
X                               return(FALSE);
X                       }
X                       if (lnum == linenum[c])
X                               ipc->flags.inrange = FALSE;
X               }
X               else if (match(p2, 0))
X                       ipc->flags.inrange = FALSE;
X       }
X       else if (*p1 == CEND)
X       {
X               if (!lastline)
X               {
X                       if (ipc->flags.allbut)
X                               return(TRUE);
X                       ipc++;
X                       return(FALSE);
X               }
```

```
X        }
X        else if (*p1 == CLNUM)
X        {
X                c = p1[1];
X                if (lnum != linenum[c])
X                {
X                        if (ipc->flags.allbut)
X                                return(TRUE);
X                        ipc++;
X                        return(FALSE);
X                }
X                if (p2)
X                        ipc->flags.inrange = TRUE;
X        }
X        else if (match(p1, 0))
X        {
X                if (p2)
X                        ipc->flags.inrange = TRUE;
X        }
X        else
X        {
X                if (ipc->flags.allbut)
X                        return(TRUE);
X                ipc++;
X                return(FALSE);
X        }
X        return(TRUE);      /* SAJ for Atari ST, Mark Williams C */
X}
X
Xstatic int match(expbuf, gf)    /* uses genbuf */
X/* match RE at expbuf against linebuf; if gf set, copy linebuf from genbuf */
Xchar    *expbuf;
X{
X        register char   *p1, *p2, c;
X
X        if (gf)
X        {
X                if (*expbuf)
X                        return(FALSE);
X                p1 = linebuf; p2 = genbuf;
X                while (*p1++ = *p2++);
X                locs = p1 = loc2;
X        }
X        else
X        {
X                p1 = linebuf;
X                locs = FALSE;
X        }
X
X        p2 = expbuf;
X        if (*p2++)
X        {
X                loc1 = p1;
```

```
X                if(*p2 == CCHR && p2[1] != *p1) /* 1st char is wrong */
X                        return(FALSE);          /*   so fail */
X                return(advance(p1, p2));        /* else try to match rest */
X        }
X
X        /* quick check for 1st character if it's literal */
X        if (*p2 == CCHR)
X        {
X                c = p2[1];              /* pull out character to search for */
X                do {
X                        if (*p1 != c)
X                                continue;       /* scan the source string */
X                        if (advance(p1, p2))    /* found it, match the rest */
X                                return(loc1 = p1, 1);
X                } while
X                        (*p1++);
X                return(FALSE);          /* didn't find that first char */
X        }
X
X        /* else try for unanchored match of the pattern */
X        do {
X                if (advance(p1, p2))
X                        return(loc1 = p1, 1);
X        } while
X                (*p1++);
X
X        /* if got here, didn't match either way */
X        return(FALSE);
X}
X
Xstatic int advance(lp, ep)
X/* attempt to advance match pointer by one pattern element */
Xregister char   *lp;            /* source (linebuf) ptr */
Xregister char   *ep;            /* regular expression element ptr */
X{
X        register char   *curlp;         /* save ptr for closures */
X        char            c;              /* scratch character holder */
X        char            *bbeg;
X        int             ct;
X
X        for (;;)
X                switch (*ep++)
X                {
X                case CCHR:              /* literal character */
X                        if (*ep++ == *lp++)     /* if chars are equal */
X                                continue;       /* matched */
X                        return(FALSE);          /* else return false */
X
X                case CDOT:              /* anything but newline */
X                        if (*lp++)              /* first NUL is at EOL */
X                                continue;       /* keep going if didn't find */
X                        return(FALSE);          /* else return false */
X
```

```
X               case CNL:               /* start-of-line */
X               case CDOL:              /* end-of-line */
X                       if (*lp == 0)           /* found that first NUL? */
X                               continue;       /* yes, keep going */
X                       return(FALSE);          /* else return false */
X
X               case CEOF:              /* end-of-address mark */
X                       loc2 = lp;              /* set second loc */
X                       return(TRUE);           /* return true */
X
X               case CCL:               /* a closure */
X                       c = *lp++ & 0177;
X                       if (ep[c>>3] & bits[c & 07])    /* is char in set? */
X                       {
X                               ep += 16;       /* then skip rest of bitmask */
X                               continue;       /*   and keep going */
X                       }
X                       return(FALSE);          /* else return false */
X
X               case CBRA:              /* start of tagged pattern */
X                       brastart[*ep++] = lp;   /* mark it */
X                       continue;               /* and go */
X
X               case CKET:              /* end of tagged pattern */
X                       bracend[*ep++] = lp;    /* mark it */
X                       continue;               /* and go */
X
X               case CBACK:
X                       bbeg = brastart[*ep];
X                       ct = bracend[*ep++] - bbeg;
X
X                       if (memcmp(bbeg, lp, ct))
X                       {
X                               lp += ct;
X                               continue;
X                       }
X                       return(FALSE);
X
X               case CBACK|STAR:
X                       bbeg = brastart[*ep];
X                       ct = bracend[*ep++] - bbeg;
X                       curlp = lp;
X                       while(memcmp(bbeg, lp, ct))
X                               lp += ct;
X
X                       while(lp >= curlp)
X                       {
X                               if (advance(lp, ep))
X                                       return(TRUE);
X                               lp -= ct;
X                       }
X                       return(FALSE);
X
```

```
X
X               case CDOT|STAR:         /* match .* */
X                       curlp = lp;             /* save closure start loc */
X                       while (*lp++);          /* match anything */
X                       goto star;              /* now look for followers */
X
X               case CCHR|STAR:         /* match <literal char>* */
X                       curlp = lp;             /* save closure start loc */
X                       while (*lp++ == *ep);   /* match many of that char */
X                       ep++;                   /* to start of next element */
X                       goto star;              /* match it and followers */
X
X               case CCL|STAR:          /* match [...]* */
X                       curlp = lp;             /* save closure start loc */
X                       do {
X                               c = *lp++ & 0x7F;       /* match any in set */
X                       } while
X                               (ep[c>>3] & bits[c & 07]);
X                       ep += 16;               /* skip past the set */
X                       goto star;              /* match followers */
X
X               star:           /* the recursion part of a * or + match */
X                       if (--lp == curlp)      /* 0 matches */
X                               continue;
X
X                       if (*ep == CCHR)
X                       {
X                               c = ep[1];
X                               do {
X                                       if (*lp != c)
X                                               continue;
X                                       if (advance(lp, ep))
X                                               return(TRUE);
X                               } while
X                                       (lp-- > curlp);
X                               return(FALSE);
X                       }
X
X                       if (*ep == CBACK)
X                       {
X                               c = *(brastart[ep[1]]);
X                               do {
X                                       if (*lp != c)
X                                               continue;
X                                       if (advance(lp, ep))
X                                               return(TRUE);
X                               } while
X                                       (lp-- > curlp);
X                               return(FALSE);
X                       }
X
X                       do {
X                               if (lp == locs)
```

```
X                                       break;
X                               if (advance(lp, ep))
X                                       return(TRUE);
X                       } while
X                               (lp-- > curlp);
X                       return(FALSE);
X
X               default:
X                       fprintf(stderr, "sed: RE error, %o\n", *--ep);
X               }
X}
X
Xstatic int substitute(ipc)
X/* perform s command */
Xsedcmd  *ipc;                          /* ptr to s command struct */
X{
X       void dosub();                   /* for if we find a match */
X
X       if (match(ipc->u.lhs, 0))               /* if no match */
X               dosub(ipc->rhs);                /* perform it once */
X       else
X               return(FALSE);                  /* command fails */
X
X       if (ipc->flags.global)                  /* if global flag enabled */
X               while(*loc2)                    /* cycle through possibles */
X                       if (match(ipc->u.lhs, 1))       /* found another */
X                               dosub(ipc->rhs);        /* so substitute */
X                       else                            /* otherwise, */
X                               break;                  /* we're done */
X       return(TRUE);                           /* we succeeded */
X}
X
Xstatic void dosub(rhsbuf)              /* uses linebuf, genbuf, spend */
X/* generate substituted right-hand side (of s command) */
Xchar    *rhsbuf;        /* where to put the result */
X{
X       register char   *lp, *sp, *rp;
X       int             c;
X       char            *place();
X
X       /* copy linebuf to genbuf up to location  1 */
X       lp = linebuf; sp = genbuf;
X       while (lp < loc1) *sp++ = *lp++;
X
X       for (rp = rhsbuf; c = *rp++; )
X       {
X               if (c == '&')
X               {
X                       sp = place(sp, loc1, loc2);
X                       continue;
X               }
X               else if (c & 0200 && (c &= 0177) >= '1' && c < MAXTAGS+'1')
X               {
```

```
X                       sp = place(sp, brastart[c-'1'], bracend[c-'1']);
X                       continue;
X               }
X               *sp++ = c & 0177;
X               if (sp >= genbuf + MAXBUF)
X                       fprintf(stderr, LTLMSG);
X       }
X       lp = loc2;
X/* MRY  loc2 = sp - genbuf + linebuf; */
X       loc2 = sp - (genbuf - linebuf);
X       while (*sp++ = *lp++)
X               if (sp >= genbuf + MAXBUF)
X                       fprintf(stderr, LTLMSG);
X       lp = linebuf; sp = genbuf;
X       while (*lp++ = *sp++);
X       spend = lp-1;
X}
X
Xstatic char *place(asp, al1, al2)               /* uses genbuf */
X/* place chars at *al1...*(al1 - 1) at asp... in genbuf[] */
Xregister char   *asp, *al1, *al2;
X{
X       while (al1 < al2)
X       {
X               *asp++ = *al1++;
X               if (asp >= genbuf + MAXBUF)
X                       fprintf(stderr, LTLMSG);
X       }
X       return(asp);
X}
X
Xstatic void listto(p1, fp)
X/* write a hex dump expansion of *p1... to fp */
Xregister char   *p1;            /* the source */
XFILE           *fp;            /* output stream to write to */
X{
X       p1--;
X       while(*p1++)
X               if (isprint(*p1))
X                       putc(*p1, fp);          /* pass it through */
X               else
X               {
X                       putc('\134', fp);               /* emit a backslash */
X                       switch(*p1)
X                       {
X                       case '\10':     putc('b', fp); break;   /* BS */
X                       case '\11':     putc('t', fp); break;   /* TAB */
X/* \11 was \9 --MRY */
X                       case '\12':     putc('n', fp); break;   /* NL */
X                       case '\15':     putc('r', fp); break;   /* CR */
X                       case '\33':     putc('e', fp); break;   /* ESC */
X                       default:        fprintf(fp, "%02x", *p1 & 0xFF);
X                       }
```

```
X                }
X        putc('\n', fp);
X}
X
Xstatic void command(ipc)
X/* execute compiled command pointed at by ipc */
Xsedcmd  *ipc;
X{
X        static int      didsub;                 /* true if last s succeeded */
X        static char     holdsp[MAXHOLD];        /* the hold space */
X        static char     *hspend = holdsp;       /* hold space end pointer */
X        register char   *p1, *p2, *p3;
X        register int    i;
X        char            *execp;
X
X        switch(ipc->command)
X        {
X        case ACMD:              /* append */
X                *aptr++ = ipc;
X                if (aptr >= appends + MAXAPPENDS)
X                        fprintf(stderr,
X                                "sed: too many appends after line %ld\n",
X                                lnum);
X                *aptr = 0;
X                break;
X
X        case CCMD:              /* change pattern space */
X                delete = TRUE;
X                if (!ipc->flags.inrange || lastline)
X                        printf("%s\n", ipc->u.lhs);
X                break;
X
X        case DCMD:              /* delete pattern space */
X                delete++;
X                break;
X
X        case CDCMD:             /* delete a line in hold space */
X                p1 = p2 = linebuf;
X                while(*p1 != '\n')
X                        if (delete = (*p1++ == 0))
X                                return;
X                p1++;
X                while(*p2++ = *p1++) continue;
X                spend = p2-1;
X                jump++;
X                break;
X
X        case EQCMD:             /* show current line number */
X                fprintf(stdout, "%ld\n", lnum);
X                break;
X
X        case GCMD:              /* copy hold space to pattern space */
X                p1 = linebuf;   p2 = holdsp;    while(*p1++ = *p2++);
```

```
X                spend = p1-1;
X                break;
X
X        case CGCMD:             /* append hold space to pattern space */
X                *spend++ = '\n';
X                p1 = spend;     p2 = holdsp;
X                while(*p1++ = *p2++)
X                        if (p1 >= linebuf + MAXBUF)
X                                break;
X                spend = p1-1;
X                break;
X
X        case HCMD:              /* copy pattern space to hold space */
X                p1 = holdsp;    p2 = linebuf;   while(*p1++ = *p2++);
X                hspend = p1-1;
X                break;
X
X        case CHCMD:             /* append pattern space to hold space */
X                *hspend++ = '\n';
X                p1 = hspend;    p2 = linebuf;
X                while(*p1++ = *p2++)
X                        if (p1 >= holdsp + MAXBUF)
X                                break;
X                hspend = p1-1;
X                break;
X
X        case ICMD:              /* insert text */
X                printf("%s\n", ipc->u.lhs);
X                break;
X
X        case BCMD:              /* branch to label */
X                jump = TRUE;
X                break;
X
X        case LCMD:              /* list text */
X                listto(linebuf, (ipc->fout != NULL)?ipc->fout:stdout); break;
X
X        case NCMD:      /* read next line into pattern space */
X                if (!nflag)
X                        puts(linebuf);  /* flush out the current line */
X                if (aptr > appends)
X                        readout();      /* do pending a, r commands */
X                if ((execp = getline(linebuf)) == BAD)
X                {
X                        pending = ipc;
X                        delete = TRUE;
X                        break;
X                }
X                spend = execp;
X                break;
X
X        case CNCMD:     /* append next line to pattern space */
X                if (aptr > appends)
```

```
X                       readout();
X               *spend++ = '\n';
X               if ((execp = getline(spend)) == BAD)
X               {
X                       pending = ipc;
X                       delete = TRUE;
X                       break;
X               }
X               spend = execp;
X               break;
X
X       case PCMD:              /* print pattern space */
X               puts(linebuf);
X               break;
X
X       case CPCMD:             /* print one line from pattern space */
X               cpcom:          /* so s command can jump here */
X               for(p1 = linebuf; *p1 != '\n' && *p1 != '\0'; )
X                       putc(*p1++, stdout);
X               putc('\n', stdout);
X               break;
X
X       case QCMD:              /* quit the stream editor */
X               if (!nflag)
X                       puts(linebuf);  /* flush out the current line */
X               if (aptr > appends)
X                       readout();      /* do any pending a and r commands */
X               exit(0);
X
X       case RCMD:              /* read a file into the stream */
X               *aptr++ = ipc;
X               if (aptr >= appends + MAXAPPENDS)
X                       fprintf(stderr,
X                               "sed: too many reads after line %ld\n",
X                               lnum);
X               *aptr = 0;
X               break;
X
X       case SCMD:              /* substitute RE */
X               didsub = substitute(ipc);
X               if (ipc->flags.print && didsub)
X                       if (ipc->flags.print == TRUE)
X                               puts(linebuf);
X                       else
X                               goto cpcom;
X               if (didsub && ipc->fout)
X                       fprintf(ipc->fout, "%s\n", linebuf);
X               break;
X
X       case TCMD:              /* branch on last s successful */
X       case CTCMD:             /* branch on last s failed */
X               if (didsub == (ipc->command == CTCMD))
X                       break;          /* no branch if last s failed, else */
```

```
X                didsub = FALSE;
X                jump = TRUE;            /*  set up to jump to assoc'd label */
X                break;
X
X        case CWCMD:             /* write one line from pattern space */
X                for(p1 = linebuf; *p1 != '\n' && *p1 != '\0'; )
X                        putc(*p1++, ipc->fout);
X                putc('\n', ipc->fout);
X                break;
X
X        case WCMD:              /* write pattern space to file */
X                fprintf(ipc->fout, "%s\n", linebuf);
X                break;
X
X        case XCMD:              /* exchange pattern and hold spaces */
X                p1 = linebuf;   p2 = genbuf;    while(*p2++ = *p1++) continue;
X                p1 = holdsp;    p2 = linebuf;   while(*p2++ = *p1++) continue;
X                spend = p2 - 1;
X                p1 = genbuf;    p2 = holdsp;    while(*p2++ = *p1++) continue;
X                hspend = p2 - 1;
X                break;
X
X        case YCMD:
X                p1 = linebuf;   p2 = ipc->u.lhs;
X                while(*p1 = p2[*p1])
X                        p1++;
X                break;
X        }
X}
X
Xstatic char *getline(buf)
X/* get next line of text to be filtered */
Xregister char   *buf;           /* where to send the input */
X{
X        if (gets(buf) != NULL)
X        {
X                lnum++;                 /* note that we got another line */
X                while(*buf++);          /* find the end of the input */
X                return(--buf);          /* return ptr to terminating null */
X        }
X        else
X        {
X                if (eargc == 0)                 /* if no more args */
X                        lastline = TRUE;        /*    set a flag */
X                return(BAD);
X        }
X}
X
Xstatic int memcmp(a, b, count)
X/* return TRUE if *a... == *b... for count chars, FALSE otherwise */
Xregister char   *a, *b;
X{
X        while(count--)                  /* look at count characters */
```

```
X                if (*a++ != *b++)       /* if any are nonequal   */
X                        return(FALSE);  /*    return FALSE for false */
X        return(TRUE);                   /* compare succeeded */
X}
X
Xstatic void readout()
X/* write file indicated by r command to output */
X{
X        register char   *p1;    /* character-fetching dummy */
X        register int    t;      /* hold input char or EOF */
X        FILE            *fi;    /* ptr to file to be read */
X
X        aptr = appends - 1;     /* arrange for pre-increment to work right */
X        while(*++aptr)
X                if ((*aptr)->command == ACMD)           /* process "a" cmd */
X                        printf("%s\n", (*aptr)->u.lhs);
X                else                                    /* process "r" cmd */
X                {
X                        if ((fi = fopen((*aptr)->u.lhs, "r")) == NULL)
X                                continue;
X                        while((t = getc(fi)) != EOF)
X                                putc((char) t, stdout);
X                        fclose(fi);
X                }
X        aptr = appends;         /* reset the append ptr */
X        *aptr = 0;
X}
X
X/* sedexec.c ends here */
END_OF_sedexec.c
if test 28811 -ne `wc -c <sedexec.c`; then
    echo shar: \"sedexec.c\" unpacked with wrong size!
fi
# end of overwriting check
fi
if test -f sedtest.bat -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"sedtest.bat\"
else
echo shar: Extracting \"sedtest.bat\" \(21 characters\)
sed "s/^X//" >sedtest.bat <<'END_OF_sedtest.bat'
Xsed -f ctrans pascal
END_OF_sedtest.bat
if test 21 -ne `wc -c <sedtest.bat`; then
    echo shar: \"sedtest.bat\" unpacked with wrong size!
fi
# end of overwriting check
fi
echo shar: End of shell archive.
exit 0


From 6708558170501425857
X-Google-Language: ENGLISH,ASCII-7-bit
```

X-Google-Thread: 100493,60608d21bbb1c293,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-02-24 11:58:40 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i017: sort -- UNIX-style sort utility part01/02
Keywords: shar
Message-ID: <6466@saturn.ucsc.edu>
Date: 24 Feb 89 19:58:40 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1416
Approved: koreth@ssyx.ucsc.edu

Submitted-by: leo@philmds.dts.philips.nl (Leo de Wit)
Posting-number: Volume 2, Issue 17
Archive-name: sort/part01

This sort program will appear familiar to *nix freaks. You can use it
to sort and/or merge multiple input files on several keys (numeric,
lexicographic). All hardware configurations should be supported (at
least I don't think it is dependent on resolution, memory size, disk
type etc.). Binary & docs submitted to comp.binaries.atari.st.

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
# This is part 1 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
#
#    Run the following text with /bin/sh to create:
#      fastcmp.asm
#      readme
#      sort.c
#      sort.doc
#      sortcomp.c
#      sortcomp.h
#      sortfile.c
#      sortfile.h
#      sortmain.c
#      sortmain.h
#
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
sed 's/^X//' << 'SHAR_EOF' > fastcmp.asm &&
X*************************************************************************
X*                                                                       *
X*   fastcmp.asm version 1.0 of 22 Januari 1989    (C) L.J.M. de Wit 1989     *
X*                                                                       *
X* This software may be used and distributed freely if not used commercially  *
```

```
X* and the originator (me) is mentioned in the source (just leave this 9 line *
X* header intact).                                                            *
X*                                                                           *
X*****************************************************************************
X*
X* fastcmp.asm: fast sort functions
X*
X*   The code here presented is offered as a replacement for some of the C
X*   functions present in sortcomp.c. Since comparisions are done a lot while
X*   sorting, some performance tweaking didn't seem misplaced.
X*
X*   As for documentation, a reference is made to the corresponding C functions
X*   in sortcomp.c. Most of the code here does simple work, like swapping
X*   arguments (for the 'r' options), and/or skipping white space (for the 'b'
X*   options; marked by 'skips' labels).
X*
X
X   section s.ccode
X
Xtab   equ   9
X
X   xref     c_df
X   xref     c_d
X   xref     c_if
X   xref     c_i
X   xref     c_af
X
X   xref     c_dfu
X   xref     c_du
X   xref     c_ifu
X   xref     c_iu
X   xref     c_afu
X
X   xdef     c_dbfr
Xc_dbfr
X* Dict/Blank/Fold/Reverse  *
X   move.l   4(sp),a0
X   move.l   8(sp),4(sp)
X   move.l   a0,8(sp)
X   move.l   12(sp),a0
X   move.l   16(sp),12(sp)
X   move.l   a0,16(sp)
X*  FALLTHROUGH
X
X   xdef     c_dbf
Xc_dbf
X* Dict/Blank/Fold *
X   movea.l  4(sp),a0
X   movea.l  8(sp),a1
Xskips0a
X   move.b   (a0)+,d0
X   cmp.b    #' ',d0
X   beq.s    skips0a
```

```
X   cmp.b    #tab,d0
X   beq.s    skips0a
X   subq.l   #1,a0
Xskips1a
X   move.b   (a1)+,d1
X   cmp.b    #' ',d1
X   beq.s    skips1a
X   cmp.b    #tab,d1
X   beq.s    skips1a
X   subq.l   #1,a1
X   move.l   a0,4(sp)
X   move.l   a1,8(sp)
X   jmp      c_df
X
X   xdef     c_dbr
Xc_dbr
X* Dict/Blank/Reverse  *
X   move.l   4(sp),a0
X   move.l   8(sp),4(sp)
X   move.l   a0,8(sp)
X   move.l   12(sp),a0
X   move.l   16(sp),12(sp)
X   move.l   a0,16(sp)
X*  FALLTHROUGH
X
X   xdef     c_db
Xc_db
X* Dict/Blank *
X   movea.l  4(sp),a0
X   movea.l  8(sp),a1
Xskips0b
X   move.b   (a0)+,d0
X   cmp.b    #' ',d0
X   beq.s    skips0b
X   cmp.b    #tab,d0
X   beq.s    skips0b
X   subq.l   #1,a0
Xskips1b
X   move.b   (a1)+,d1
X   cmp.b    #' ',d1
X   beq.s    skips1b
X   cmp.b    #tab,d1
X   beq.s    skips1b
X   subq.l   #1,a1
X   move.l   a0,4(sp)
X   move.l   a1,8(sp)
X   jmp      c_d
X
X   xdef     c_dfr
Xc_dfr
X* Dict/Fold/Reverse  *
X   move.l   4(sp),a0
X   move.l   8(sp),4(sp)
```

```
X   move.l   a0,8(sp)
X   move.l   12(sp),a0
X   move.l   16(sp),12(sp)
X   move.l   a0,16(sp)
X   jmp      c_df
X
X   xdef     c_dr
Xc_dr
X* Dict/Reverse  *
X   move.l   4(sp),a0
X   move.l   8(sp),4(sp)
X   move.l   a0,8(sp)
X   move.l   12(sp),a0
X   move.l   16(sp),12(sp)
X   move.l   a0,16(sp)
X   jmp      c_d
X
X   xdef     c_ibfr
Xc_ibfr
X* Norm/Blank/Fold/Reverse  *
X   move.l   4(sp),a0
X   move.l   8(sp),4(sp)
X   move.l   a0,8(sp)
X   move.l   12(sp),a0
X   move.l   16(sp),12(sp)
X   move.l   a0,16(sp)
X*  FALLTHROUGH
X
X   xdef     c_ibf
Xc_ibf
X* Norm/Blank/Fold *
X   movea.l  4(sp),a0
X   movea.l  8(sp),a1
Xskips0c
X   move.b   (a0)+,d0
X   cmp.b    #' ',d0
X   beq.s    skips0c
X   cmp.b    #tab,d0
X   beq.s    skips0c
X   subq.l   #1,a0
Xskips1c
X   move.b   (a1)+,d1
X   cmp.b    #' ',d1
X   beq.s    skips1c
X   cmp.b    #tab,d1
X   beq.s    skips1c
X   subq.l   #1,a1
X   move.l   a0,4(sp)
X   move.l   a1,8(sp)
X   jmp      c_if
X
X   xdef     c_ibr
Xc_ibr
```

```
X* Norm/Blank/Reverse  *
X   move.l   4(sp),a0
X   move.l   8(sp),4(sp)
X   move.l   a0,8(sp)
X   move.l   12(sp),a0
X   move.l   16(sp),12(sp)
X   move.l   a0,16(sp)
X*  FALLTHROUGH
X
X   xdef     c_ib
Xc_ib
X* Norm/Blank *
X   movea.l  4(sp),a0
X   movea.l  8(sp),a1
Xskips0d
X   move.b   (a0)+,d0
X   cmp.b    #' ',d0
X   beq.s    skips0d
X   cmp.b    #tab,d0
X   beq.s    skips0d
X   subq.l   #1,a0
Xskips1d
X   move.b   (a1)+,d1
X   cmp.b    #' ',d1
X   beq.s    skips1d
X   cmp.b    #tab,d1
X   beq.s    skips1d
X   subq.l   #1,a1
X   move.l   a0,4(sp)
X   move.l   a1,8(sp)
X   jmp      c_i
X
X   xdef     c_ifr
Xc_ifr
X* Norm/Fold/Reverse  *
X   move.l   4(sp),a0
X   move.l   8(sp),4(sp)
X   move.l   a0,8(sp)
X   move.l   12(sp),a0
X   move.l   16(sp),12(sp)
X   move.l   a0,16(sp)
X   jmp      c_if
X
X   xdef     c_ir
Xc_ir
X* Norm/Reverse  *
X   move.l   4(sp),a0
X   move.l   8(sp),4(sp)
X   move.l   a0,8(sp)
X   move.l   12(sp),a0
X   move.l   16(sp),12(sp)
X   move.l   a0,16(sp)
X   jmp      c_i
```

```
X
X   xdef     c_abfr
Xc_abfr
X* Blank/Fold/Reverse  *
X   move.l   4(sp),a0
X   move.l   8(sp),4(sp)
X   move.l   a0,8(sp)
X   move.l   12(sp),a0
X   move.l   16(sp),12(sp)
X   move.l   a0,16(sp)
X*  FALLTHROUGH
X
X   xdef     c_abf
Xc_abf
X* Blank/Fold *
X   movea.l  4(sp),a0
X   movea.l  8(sp),a1
Xskips0e
X   move.b   (a0)+,d0
X   cmp.b    #' ',d0
X   beq.s    skips0e
X   cmp.b    #tab,d0
X   beq.s    skips0e
X   subq.l   #1,a0
Xskips1e
X   move.b   (a1)+,d1
X   cmp.b    #' ',d1
X   beq.s    skips1e
X   cmp.b    #tab,d1
X   beq.s    skips1e
X   subq.l   #1,a1
X   move.l   a0,4(sp)
X   move.l   a1,8(sp)
X   jmp      c_af
X
X   xdef     c_abr
Xc_abr
X* Blank/Reverse  *
X   move.l   4(sp),a0
X   move.l   8(sp),4(sp)
X   move.l   a0,8(sp)
X   move.l   12(sp),a0
X   move.l   16(sp),12(sp)
X   move.l   a0,16(sp)
X*  FALLTHROUGH
X
X   xdef     c_ab
Xc_ab
X* Blank *
X   movea.l  4(sp),a0
X   movea.l  8(sp),a1
Xskips0f
X   move.b   (a0)+,d0
```

```
X   cmp.b    #' ',d0
X   beq.s    skips0f
X   cmp.b    #tab,d0
X   beq.s    skips0f
X   subq.l   #1,a0
Xskips1f
X   move.b   (a1)+,d1
X   cmp.b    #' ',d1
X   beq.s    skips1f
X   cmp.b    #tab,d1
X   beq.s    skips1f
X   subq.l   #1,a1
X   move.l   a0,4(sp)
X   move.l   a1,8(sp)
X   jmp      c_a
X
X   xdef     c_afr
Xc_afr
X* Fold/Reverse  *
X   move.l   4(sp),a0
X   move.l   8(sp),4(sp)
X   move.l   a0,8(sp)
X   move.l   12(sp),a0
X   move.l   16(sp),12(sp)
X   move.l   a0,16(sp)
X   jmp      c_af
X
X   xdef     c_ar
Xc_ar
X* Reverse  *
X   move.l   4(sp),a0
X   move.l   8(sp),4(sp)
X   move.l   a0,8(sp)
X   move.l   12(sp),a0
X   move.l   16(sp),12(sp)
X   move.l   a0,16(sp)
X   jmp      c_a
X
X   xdef     c_a
Xc_a
X   move.l   4(sp),d0
X   move.l   12(sp),d1
X   sub.l    d0,d1
X   move.l   d0,a0
X   move.l   8(sp),d0
X   move.l   16(sp),d2
X   sub.l    d0,d2
X   move.l   d0,a1
X   move.w   d2,d0
X   sub.w    d1,d0
X   blt.s    c_asmall
X   move.w   d1,d2
Xc_asmall
```

```
X   dbra     d2,c_aloop
Xc_aeqs
X   ext.l    d0
X   rts
X
Xc_aloop
X   cmp.b    (a0)+,(a1)+
X   dbne     d2,c_aloop
X   beq.s    c_aeqs
X   bpl.s    c_aplus
X   moveq.l  #1,d0
X   rts
Xc_aplus
X   moveq.l  #-1,d0
X   rts
X
X* Unbounded comparisions start here *
X
X   xdef     c_dbfru
Xc_dbfru
X* Dict/Blank/Fold/Reverse  *
X   move.l   4(sp),a1
X   move.l   8(sp),a0
X   bra.s    skips0g
X
X   xdef     c_dbfu
Xc_dbfu
X* Dict/Blank/Fold  *
X   movea.l  4(sp),a0
X   movea.l  8(sp),a1
Xskips0g
X   move.b   (a0)+,d0
X   cmp.b    #' ',d0
X   beq.s    skips0g
X   cmp.b    #tab,d0
X   beq.s    skips0g
X   subq.l   #1,a0
Xskips1g
X   move.b   (a1)+,d1
X   cmp.b    #' ',d1
X   beq.s    skips1g
X   cmp.b    #tab,d1
X   beq.s    skips1g
X   subq.l   #1,a1
X   move.l   a0,4(sp)
X   move.l   a1,8(sp)
X   jmp      c_dfu
X
X   xdef     c_dbru
Xc_dbru
X* Dict/Blank/Reverse  *
X   move.l   4(sp),a1
X   move.l   8(sp),a0
```

```
X   bra.s    skips0h
X
X   xdef     c_dbu
Xc_dbu
X* Dict/Blank  *
X   movea.l  4(sp),a0
X   movea.l  8(sp),a1
Xskips0h
X   move.b   (a0)+,d0
X   cmp.b    #' ',d0
X   beq.s    skips0h
X   cmp.b    #tab,d0
X   beq.s    skips0h
X   subq.l   #1,a0
Xskips1h
X   move.b   (a1)+,d1
X   cmp.b    #' ',d1
X   beq.s    skips1h
X   cmp.b    #tab,d1
X   beq.s    skips1h
X   subq.l   #1,a1
X   move.l   a0,4(sp)
X   move.l   a1,8(sp)
X   jmp      c_du
X
X   xdef     c_dfru
Xc_dfru
X* Dict/Fold/Reverse  *
X   move.l   4(sp),a0
X   move.l   8(sp),4(sp)
X   move.l   a0,8(sp)
X   jmp      c_dfu
X
X   xdef     c_dru
Xc_dru
X* Dict/Reverse  *
X   move.l   4(sp),a0
X   move.l   8(sp),4(sp)
X   move.l   a0,8(sp)
X   jmp      c_du
X
X   xdef     c_ibfru
Xc_ibfru
X* Norm/Blank/Fold/Reverse  *
X   move.l   4(sp),a1
X   move.l   8(sp),a0
X   bra.s    skips0i
X*  FALLTHROUGH
X
X   xdef     c_ibfu
Xc_ibfu
X* Norm/Blank/Fold  *
X   movea.l  4(sp),a0
```

```
X   movea.l  8(sp),a1
Xskips0i
X   move.b   (a0)+,d0
X   cmp.b    #' ',d0
X   beq.s    skips0i
X   cmp.b    #tab,d0
X   beq.s    skips0i
X   subq.l   #1,a0
Xskips1i
X   move.b   (a1)+,d1
X   cmp.b    #' ',d1
X   beq.s    skips1i
X   cmp.b    #tab,d1
X   beq.s    skips1i
X   subq.l   #1,a1
X   move.l   a0,4(sp)
X   move.l   a1,8(sp)
X   jmp      c_ifu
X
X   xdef     c_ibru
Xc_ibru
X* Norm/Blank/Reverse  *
X   move.l   4(sp),a1
X   move.l   8(sp),a0
X   bra.s    skips0j
X
X   xdef     c_ibu
Xc_ibu
X* Norm/Blank  *
X   movea.l  4(sp),a0
X   movea.l  8(sp),a1
Xskips0j
X   move.b   (a0)+,d0
X   cmp.b    #' ',d0
X   beq.s    skips0j
X   cmp.b    #tab,d0
X   beq.s    skips0j
X   subq.l   #1,a0
Xskips1j
X   move.b   (a1)+,d1
X   cmp.b    #' ',d1
X   beq.s    skips1j
X   cmp.b    #tab,d1
X   beq.s    skips1j
X   subq.l   #1,a1
X   move.l   a0,4(sp)
X   move.l   a1,8(sp)
X   jmp      c_iu
X
X   xdef     c_ifru
Xc_ifru
X* Norm/Fold/Reverse  *
X   move.l   4(sp),a0
```

```
X   move.l   8(sp),4(sp)
X   move.l   a0,8(sp)
X   jmp      c_ifu
X
X   xdef     c_iru
Xc_iru
X* Norm/Reverse  *
X   move.l   4(sp),a0
X   move.l   8(sp),4(sp)
X   move.l   a0,8(sp)
X   jmp      c_iu
X
X   xdef     c_abfru
Xc_abfru
X* Blank/Fold/Reverse  *
X   move.l   4(sp),a1
X   move.l   8(sp),a0
X   bra.s    skips0k
X
X   xdef     c_abfu
Xc_abfu
X* Blank/Fold  *
X   movea.l  4(sp),a0
X   movea.l  8(sp),a1
Xskips0k
X   move.b   (a0)+,d0
X   cmp.b    #' ',d0
X   beq.s    skips0k
X   cmp.b    #tab,d0
X   beq.s    skips0k
X   subq.l   #1,a0
Xskips1k
X   move.b   (a1)+,d1
X   cmp.b    #' ',d1
X   beq.s    skips1k
X   cmp.b    #tab,d1
X   beq.s    skips1k
X   subq.l   #1,a1
X   move.l   a0,4(sp)
X   move.l   a1,8(sp)
X   jmp      c_afu
X
X   xdef     c_abru
Xc_abru
X* Blank/Reverse  *
X   move.l   4(sp),a1
X   move.l   8(sp),a0
X   bra.s    skips0l
X
X   xdef     c_abu
Xc_abu
X* Blank  *
X   movea.l  4(sp),a0
```

```
X   movea.l  8(sp),a1
Xskips0l
X   move.b   (a0)+,d0
X   cmp.b    #' ',d0
X   beq.s    skips0l
X   cmp.b    #tab,d0
X   beq.s    skips0l
X   subq.l   #1,a0
Xskips1l
X   move.b   (a1)+,d1
X   cmp.b    #' ',d1
X   beq.s    skips1l
X   cmp.b    #tab,d1
X   beq.s    skips1l
X   subq.l   #1,a1
X   move.l   a0,4(sp)
X   move.l   a1,8(sp)
X   jmp      c_au
X
X   xdef     c_afru
Xc_afru
X* Fold/Reverse  *
X   move.l   4(sp),a0
X   move.l   8(sp),4(sp)
X   move.l   a0,8(sp)
X   jmp      c_afu
X
X   xdef     c_aru
Xc_aru
X* Reverse  *
X   move.l   4(sp),a1
X   move.l   8(sp),a0
X   bra.s    c_auloop
X
X   xdef     c_au
Xc_au
X   movea.l 4(sp),a0
X   movea.l 8(sp),a1
Xc_auloop
X   move.b  (a0)+,d0
X   beq.s   c_auex0
X   sub.b   (a1)+,d0
X   bne.s   c_auex1
X   move.b  (a0)+,d0
X   beq.s   c_auex0
X   sub.b   (a1)+,d0
X   bne.s   c_auex1
X   move.b  (a0)+,d0
X   beq.s   c_auex0
X   sub.b   (a1)+,d0
X   bne.s   c_auex1
X   move.b  (a0)+,d0
X   beq.s   c_auex0
```

```
X   sub.b   (a1)+,d0
X   beq.s   c_auloop
Xc_auex1
X   ext.w   d0
X   ext.l   d0
X   rts
Xc_auex0
X   sub.b   (a1),d0
X   ext.w   d0
X   ext.l   d0
X   rts
X
X   end
SHAR_EOF
chmod 0600 fastcmp.asm || echo "restore of fastcmp.asm fails"
sed 's/^X//' << 'SHAR_EOF' > readme &&
X
X                               Sort
X                               ----
X
X
X   The sort utility here presented is comparable to the Unix sort utility;
X   even most of the flags are the same. Sorting is done in core as much as
X   possible; else a \tmp directory (or a user supplied one) is used for
X   creation of temporary files.
X
X   Two speedups have been done that might affect portability:
X       1) a sortfile.c module is added that is included if the symbol BESTIO
X   is defined. The stdio buffers have been increased to 2K, which seems a
X   better choice than the standard 512.
X       2) a fastcmp.asm module is added for faster comparision functions;
X   these functions are taken from sortcomp.c if M68000 is not defined.
X
X   Contents of complete package:
X
X      fastcmp.asm    - module containing faster comparision functions
X      readme         - this introductory file
X      sort.c         - module containing basic sort functions
X      sort.doc       - manual page for sort
X      sort.prg       - the executable program
X      sortcomp.c     - module containing comparision functions
X      sortcomp.h     - header for sortcomp module
X      sortfile.c     - module containing improved I/O functions
X      sortfile.h     - header for sortfile module
X      sortmain.c     - main module; argument scanning; general key matching
X      sortmain.h     - header for sortmain module
X
X   It will probably be distributed in two chucks: one containing .c and .h
X   files, readme and sort.doc (submitted to comp.sources.atari.st) and one
X   containing sort.prg, readme and sort.doc (s. to comp.binaries.atari.st).
X
X   Compiled with Lattice C; first compiler pass with -dBESTIO option, second
X   with -r option. Fastcmp.asm was assembled using the GST linker.
```

```
X
X   This program was the result of a question on the net, asking for a disk
X   based sort. It seemed like a nice project (in fact, it was 8-), and I
X   hope at least the person who asked it can make some use of it.
X
X   For questions, bug reports etc. try
X
X      L.J.M. de Wit
X      Nachtegaallaan 7
X      5731XP Mierlo
X      Holland
X      E-mail: ..!mcvax!philmds!leo (and various other paths...)
X
X   Enjoy! (sort of ...)
X
SHAR_EOF
chmod 0600 readme || echo "restore of readme fails"
sed 's/^X//' << 'SHAR_EOF' > sort.c &&
X/*****************************************************************************
X *                                                                           *
X *    sort.c     version 1.0 of 22 Januari 1989    (C) L.J.M. de Wit 1989    *
X *                                                                           *
X * This software may be used and distributed freely if not used commercially  *
X * and the originator (me) is mentioned in the source (just leave this 9 line *
X * header intact).                                                           *
X *                                                                           *
X *****************************************************************************
X *
X * sort.c: basic sort functions
X */
X
X#include <stdio.h>
X#include <osbind.h>
X#include "sortmain.h"
X#include "sortfile.h"
X
X#define LINELEN  256                        /* Max length of input lines*/
X#define MAXNAMLEN 80                        /* Max length of filename   */
X#define MAXMERGE 8                          /* Max. no of files to be   */
X                                            /* simultaneously merged    */
Xstatic int (*cmp)();                        /* Ptr to compare function  */
X
Xstatic void sortsub(),                      /* Recursive quicksort      */
X            merge();                       /* Merge sorted files       */
Xstatic void tempname();                     /* Creates temporary name   */
Xstatic int isort();                         /* Initial sort of file(s)  */
X
Xstatic char tmpdir[MAXNAMLEN];              /* Name of temp files dir   */
X
Xstatic char insave[2][LINELEN];             /* Used by unique           */
X
Xvoid settemp(tempdir)                       /* Sets the temp files dir  */
Xchar *tempdir;
```

```
X{
X   strcpy(tmpdir,tempdir);
X}
X
Xstatic void sortsub(sbase,stop)                /* Sort the pointers        */
Xchar **sbase, **stop;                           /* between sbase and stop   */
X{
X    register char **base = sbase,              /* All below base and above */
X                  **top = stop,                /* top is kept sorted with  */
X                  **mid,                       /* respect to element *mid  */
X                  *tmp;                        /* For swapping values      */
X    register long compfunc = (long)cmp;        /* Kludge to use extra reg. */
X
X#define CMP (*(int (*)())compfunc)             /* Replaces (*cmp)          */
X
X    mid = base + ((top - base) >> 1);          /* Choose pivot             */
X    --top;
X    if (top - base >= 6) {                     /* Make sure pivot mid value*/
X        if (CMP(*base,*mid) > 0) {
X            if (CMP(*mid,*top) <= 0) {
X                if (CMP(*base,*top) > 0) {
X                    tmp = *mid;                /* Swap *mid and *top       */
X                    *mid = *top;
X                    *top = tmp;
X                } else {
X                    tmp = *mid;                /* Swap *base and *mid      */
X                    *mid = *base;
X                    *base = tmp;
X                }
X            }
X        } else if (CMP(*mid,*top) > 0) {
X            tmp = *mid;                        /* Swap *mid and *top       */
X            *mid = *top;
X            *top = tmp;
X        }
X    } else {                                   /* <= 6 elts: use bubblesort*/
X        for ( ; base < top; --top) {
X            for (mid = base; mid < top; mid++) {
X                if (CMP(*mid,mid[1]) > 0) {
X                    tmp = *mid; *mid = mid[1]; mid[1] = tmp;
X                }
X            }
X        }
X
X        return;                                /* and return ...           */
X    }
X    do {
X        for ( ; base < mid && CMP(*base,*mid) <= 0; base++) ;
X        for ( ; top > mid && CMP(*mid,*top) <= 0; --top) ;
X        if (base < top) {
X            tmp = *base; *base = *top; *top = tmp; /* Swap *base and *top   */
X            if (mid == top) {
X                mid = base;
```

```
X                --top;
X            } else if (mid == base) {
X                mid = top;
X                base++;
X            }
X        }
X    } while (base < top);                          /* Until all sorted to *mid */
X    if (mid - sbase > 1) {                         /* Sort lower half if >1 elt*/
X        sortsub(sbase,mid);
X    }
X    if (stop - mid > 1) {                          /* Sort upper half if >1 elt*/
X        sortsub(mid,stop);
X    }
X}
X
Xstatic void merge(infnames,level,
X                  startno,nmerge,outfname)      /* Merge sorted input files */
Xchar **infnames;                                  /* Input file names array   */
Xint level,                                        /* No. of times merged      */
X    startno,                                     /* Sequence no. within level*/
X    nmerge;                                      /* No. of files to merge    */
Xchar *outfname;                                   /* File to be written to    */
X{
X    static char *inbuf[MAXMERGE];                /* (0)Ptrs to input buffers */
X    static FILE *infp[MAXMERGE], *outfp;         /* Input and output FILE *'s*/
X    char namebuf[MAXNAMLEN];                     /* To store temp file names */
X    char *nameptr;
X    int cur,                                     /* Number of current file   */
X        i, j,
X        nbusy = 0,                               /* # of input files in use  */
X        tmp;
X    int order[MAXMERGE];                         /* Ordering of file numbers */
X    char *s;
X    long avail;
X
X    avail = (Malloc(-1) - 20480) & ~1;           /* Get free - 20K           */
X    avail /= nmerge + 1;
X    avail -= avail % 2048;
X
X    if (outfname == (char *)0) {
X        outfp = stdout;                          /* stdout is default        */
X    } else {
X        outfp = fopen(outfname,"w");
X        if (outfp == (FILE *)0) {
X            error("%s: cannot open for write\n",outfname);
X        }
X    }
X
X    if ((s = (char *)Malloc(avail)) == (char *)-1) { /* Grab it     */
X        error("memory allocation failed\n",(char *)0);
X    }
X    setbuffer(outfp,s,avail);
X
```

```
X    if (level < 0) {
X        if (*infnames != (char *)0) {
X            for (i = 0; i < startno; i++) {
X                infnames++;
X            }
X        }
X    }
X
X    for (i = 0; i < nmerge; i++) {              /* Open each input file     */
X        if (level < 0) {
X            nameptr = *infnames++;
X        } else {
X            tempname(namebuf,level,startno+i);
X            nameptr = namebuf;
X        }
X        if (nameptr == (char *)0 || !strcmp(nameptr,"-")) {
X            infp[i] = stdin;                    /* stdin is default         */
X        } else {
X            infp[i] = fopen(nameptr,"r");
X            if (infp[i] == (FILE *)0) {
X                error("%s: cannot open for read\n",nameptr);
X            }
X        }
X        if ((s = (char *)Malloc(avail)) == (char *)-1) { /* Grab it     */
X            error("memory allocation failed\n",(char *)0);
X        }
X        setbuffer(infp[i],s,avail);
X
X        if (inbuf[i] == (char *)0) {            /* Possibly allocate buffer */
X            if ((inbuf[i] = (char *)malloc(LINELEN)) == (char *)0) {
X                error("memory allocation failed\n",(char *)0);
X            }
X        }
X        if (fgets(inbuf[i],LINELEN,infp[i]) != (char *)0) { /* Read 1st line*/
X            order[nbusy++] = i;                 /* Enter its seq no.        */
X            for (j = nbusy - 1; j >= 1 &&       /* Put in place in order    */
X                      (*cmp)(inbuf[order[j-1]],inbuf[order[j]]) > 0; --j) {
X                tmp = order[j-1]; order[j-1] = order[j]; order[j] = tmp;
X            }
X        }
X    }
X
X    if ((options & UNIQUE) && nbusy > 0) {
X        insave[0][0] = inbuf[order[0]][0] + 1;  /* Force difference */
X    }
X
X    while (nbusy > 0) {
X        cur = order[0];                         /* No. of lowest ordered    */
X        if (options & UNIQUE) {
X            if ((*cmp)(inbuf[cur],insave[0])) { /* Differs from previous one*/
X                fputs(inbuf[cur],outfp);        /* Write out lowest ordered */
X                strcpy(insave[0],inbuf[cur]);
X            }
```

```
X        } else {
X            fputs(inbuf[cur],outfp);            /* Write out lowest ordered */
X        }
X        if (fgets(inbuf[cur],LINELEN,infp[cur]) == (char *)0) {/* No next   */
X            for (i = 1; i < nbusy; i++) {       /* Reorder rest             */
X                order[i-1] = order[i];
X            }
X            --nbusy;
X        } else {                                /* Binary search            */
X            if (nbusy > 1 && (*cmp)(inbuf[cur],inbuf[order[1]]) > 0) {
X                if (nbusy > 2) {
X                    int low, high = nbusy - 1, mid;
X
X                    if ((*cmp)(inbuf[cur],inbuf[order[high]]) >= 0) {
X                        low = high;
X                    } else {
X                        low = 1;
X                        while (high - low > 1) {
X                            mid = (high + low) >> 1;
X                            if ((*cmp)(inbuf[cur],inbuf[order[mid]]) >= 0) {
X                                low = mid;
X                            } else {
X                                high = mid;
X                            }
X                        }
X                    }
X                    for (i = 0; i < low; i++) { /* Shift elements 1..low to */
X                        order[i] = order[i+1];  /* 0..low-1 to accomodate   */
X                    }
X                    order[low] = cur;           /* for the new index cur    */
X                } else {
X                    order[0] = order[1]; order[1] = cur;
X                }
X            }
X        }
X    }
X
X    for (i = 0; i < nmerge; i++) {
X        fclose(infp[i]);                        /* Close each input file    */
X        if (level >= 0) {                       /* (also frees Malloced buf)*/
X            tempname(namebuf,level,startno+i);
X            unlink(namebuf);                    /* and delete if temp file  */
X        }
X    }
X    fclose(outfp);                              /* Close output file        */
X}
X
Xvoid allsort(infnames,destname,compar)          /* Main sort entry point    */
Xchar **infnames,                                /* Array of input filenames */
X      *destname;                                /* Destination file name    */
Xint (*compar)();                                /* Comparision function     */
X{
X    char **infs;
```

```
X    char outfname[MAXNAMLEN];                  /* For temp output filenames*/
X    int level = 0,                             /* Merge level              */
X        nfiles,                                /* No. of files to merge    */
X        startno,                               /* Order no. within nfiles  */
X        nmerge = 0;                            /* # of files to merge at a */
X                                               /* time, initialized to     */
X                                               /* hush the compiler        */
X    int sameout = 0;                           /* same in as outfile?      */
X    int merging;
X
X    cmp = compar;                              /* Set compare function used*/
X                                               /* throughout this module   */
X    nfiles = 0;
X    for (infs = infnames; *infs != (char *)0; infs++) {
X        nfiles++;
X        if (destname != (char *)0 && !strcmp(destname,*infs)) {
X            sameout = 1;
X        }
X    }
X    if (nfiles == 0) {
X        nfiles = 1;
X    }
X
X    if (options & MERGEONLY) {
X        level = -1;
X        merging = 1;
X    } else {
X        nfiles = isort(infnames,destname);     /* # Files left after incore*/
X        merging = nfiles > 1;
X        level = 0;
X    }
X
X    while (merging) {
X        merging = nfiles > MAXMERGE || (level < 0 && sameout);
X        for (startno = 0; startno < nfiles; startno += nmerge) {
X            if ((nmerge = nfiles - startno) > MAXMERGE) {
X                nmerge = MAXMERGE;             /* Use at most MAXMERGE     */
X            }
X            tempname(outfname,level+1,startno/MAXMERGE);
X            merge(infnames,level,startno,nmerge,
X                                  (merging) ? outfname : destname);
X        }
X        nfiles = 1 + (nfiles-1)/MAXMERGE;
X        level++;                               /* Next merge level */
X    }
X}
X
Xstatic void tempname(buf,level,seqno)          /* Create name of temp file */
Xchar *buf;                                      /* Will contain name formed */
Xint level, seqno;                              /* Merge level & seq. no.   */
X{
X    sprintf(buf,"%s\\sort%04d.%03d",tmpdir,level,seqno);
X}
```

```
X
Xstatic FILE *reopen(infp,fname)                /* Open input file          */
XFILE *infp;                                    /* Fileptr to possibly close*/
Xchar *fname;                                   /* File to be opened        */
X{
X    if (infp != (FILE *)0) {                   /* Close a valid file ptr   */
X        fclose(infp);
X    }
X    if (!strcmp(fname,"-")) {
X        return stdin;
X    }
X    if ((infp = fopen(fname,"r")) == (FILE *)0) {
X        error("%s: cannot open for read\n",fname);
X    }
X    return infp;
X}
X
Xstatic int isort(infnames,destname)             /* Initial in-memory sort   */
Xchar **infnames,                                /* Array of input files     */
X        *destname;                             /* File to write result to  */
X{
X    FILE *infp, *outfp;                        /* Input, output file ptr   */
X    int avail,                                 /* Available free space     */
X        seqno = 0,                             /* To count multiple writes */
X        isfree;                                /* Not still used from avail*/
X    char *sortdata,                            /* Start of usable space    */
X         *snext,                               /* First unused position    */
X         **sorttop,                            /* Top of usable space      */
X         **sortbase;                           /* Start of pointer space   */
X    char outnambuf[MAXNAMLEN],                 /* To create temp name in   */
X         *outfname,                            /* Name of output file      */
X         *fromfile;                            /* Name of input file       */
X
X    if (*infnames == (char *)0) {
X        fromfile = "standard input";
X        infp = stdin;
X    } else {
X        infp = reopen((FILE *)0,fromfile = *infnames++);
X    }
X    if (options & CHECKONLY) {                 /* Check if input ordered   */
X        int next = 1;
X
X        while (fgets(insave[0],sizeof(insave[0]),infp) == (char *)0) {
X            if (*infnames == (char *)0) {
X                exit(0);                       /* All input files empty    */
X            } else {
X                infp = reopen((FILE *)0,fromfile = *infnames++);
X            }
X        }                                      /* 1st line of 1st non-empty*/
X        do {
X            if (fgets(insave[next],sizeof(insave[next]),infp) == (char *)0) {
X                if (*infnames == (char *)0) {
X                    exit(0);                   /* Input was ordered OK     */
```

```
X               } else {
X                   infp = reopen((FILE *)0,fromfile = *infnames++);
X               }
X           } else {
X               if ((*cmp)(insave[1-next],insave[next]) > 0) { /* Compare   */
X                   error("out of order and -c specified\n",(char *)0);
X               }
X               next = 1 - next;                /* Swap buffer to be used   */
X           }
X       } while (1);
X   }
X   avail = (Malloc(-1) - 20480) & ~1;          /* Get free mem but 20K,even*/
X
X   if ((sortdata = (char *)Malloc(avail)) == (char *)-1) { /* Grab it      */
X       error("memory allocation failed\n",(char *)0);
X   }
X   sorttop = (char **)(sortdata + avail);      /* Ptrs start at top of mem,*/
X                                               /* going downwards          */
X   do {
X       snext = sortdata; isfree = avail;
X       sortbase = sorttop;
X       for ( ; isfree > LINELEN; ) {           /* Read lines while room    */
X           while (fgets(snext,isfree,infp) == (char *)0) {
X               if (*infnames == (char *)0) {
X                   break;
X               } else {
X                   infp = reopen(infp,fromfile = *infnames++);
X               }
X           }
X           if (feof(infp)) break;              /* All input read           */
X           *--sortbase = snext;                /* Store current ptr        */
X           snext += strlen(snext) + 1;         /* Skip the line read       */
X           isfree = (char *)(sortbase) - snext;
X       }
X       if (sortbase == sorttop && seqno > 0) {
X           break;                              /* No lines this time       */
X       }
X       sortsub(sortbase,sorttop);
X       tempname(outnambuf,0,seqno++);
X       outfname = outnambuf;
X       if (feof(infp)) {
X           if (*infnames == (char *)0) {
X               if (seqno == 1) {               /* All read the first time: */
X                   outfname = destname;        /* Write to file destname   */
X               }
X           } else {                            /* Open next input file     */
X               infp = reopen(infp,fromfile = *infnames++);
X           }
X       }
X       if (outfname == (char *)0) {            /* Open output file         */
X           outfp = stdout;
X       } else {
X           outfp = fopen(outfname,"w");
```

```
X            if (outfp == (FILE *)0) {
X                error("%s: cannot open for write\n",outfname);
X            }
X        }
X        if ((options & UNIQUE) && sorttop > sortbase) {
X            insave[0][0] = sortbase[0][0] + 1;     /* Force difference      */
X            for ( ; sortbase < sorttop; sortbase++) {
X                if ((*cmp)(*sortbase,insave[0])) { /* != previous          */
X                    fputs(*sortbase,outfp);
X                    strcpy(insave[0],*sortbase);
X                }
X            }
X        } else {
X            for ( ; sortbase < sorttop; sortbase++) {
X                fputs(*sortbase,outfp);
X            }
X        }
X        fclose(outfp);
X    } while (!feof(infp));
X
X    fclose(infp);                              /* Close last input file    */
X    Mfree(sortdata);                           /* Release Malloced mem     */
X
X    return seqno;                              /* No of output files used  */
X}
SHAR_EOF
chmod 0600 sort.c || echo "restore of sort.c fails"
sed 's/^X//' << 'SHAR_EOF' > sort.doc &&
X
X
X
X                                                                sort(1)
X
X
X
XNAME
X     sort - sort and/or merge files
X
XSYNTAX
X     sort [ -bdfimnrutx ] [ +pos1  [ -pos2 ] ] [ -o name ] [ -T dir ] [ name...]
X
XDESCRIPTION
X     Sort sorts, then writes the lines of the named files to its standard
X     output, or a file (-o option).
X     Default (no input files are named), the standard input is sorted.
X     The name "-" stands for the standard input, so that standard input can be
X     used together with other input files.
X
X     By default, the sort key is an entire line; ordering is lexicographic.
X     The global ordering is affected by zero or more of the following options
X     (of which some combinations are meaningless, e.g. d and n).
X
```

```
X     -b   Leading blanks (spaces and tabs) are skipped in field comparisons.
X
X     -d   Dictionary order: only letters, digits and blanks are
X          significant in comparisons.
X
X     -f   Fold: treat upper case as if lower case.
X
X     -i   Ignore characters outside the ASCII range 040-0176 in
X          nonnumeric comparisons.
X
X     -n   An initial numeric string, consisting of optional
X          blanks, optional minus sign, and zero or more digits
X          with optional decimal point, is sorted by arithmetic
X          value. This option implies option b.
X
X     -r   Reverse the result of comparisons.
X
X     -tx  `Tab character' field separator is x.
X
X     The notation +pos1 -pos2 restricts a sort key to a field
X     beginning at pos1 and ending just before pos2.
X     Pos1 and pos2 each have the form m.n, optionally followed by one
X     or more of the options bdfinr, where m tells a number of fields
X     to skip from the beginning of the line and n tells a number
X     of characters to skip further.  If any options are present
X     they override all the global ordering options for this key.
X     If the b option is in effect n is counted from the first
X     nonblank in the field.
X     A missing .n means .0; a missing -pos2 means the end of the
X     line.  Under the -tx option, fields are strings separated by
X     x; otherwise fields are nonempty nonblank strings separated
X     by blanks.
X
X     When there are multiple sort keys, later keys are compared
X     only after all earlier keys compare equal.
X
X     These are additional (global) options:
X
X     -c   Check that the input file is sorted according to the key(s);
X          if the file is out of sort, an error message is printed.
X          Exit code is 0/1 for correctly/incorrectly sorted input.
X
X     -m   Merge only, the input files are already sorted.
X
X     -o   The next argument is the output file used instead of the
X          standard output. This file may be the same as one of the inputs.
X
X     -T   Temporary files directory. Default is \tmp on the current drive.
X
X     -u   Unique: Select only one in each set of, according to the
X          key(s), equal lines.
X
XRESTRICTIONS
```

```
X     Very long (>256 chars) lines are silently truncated.
X     The line terminator sequence \r\n is present in the line when
X     doing comparisions; this should not present problems. Use of \0 in
X     input is discouraged, since this will be seen as a C string terminator.
X
XDIAGNOSTICS
X     Comments and exits with nonzero status for various trouble
X     conditions and for disorder discovered under option c(heck).
X
XFILES
X     \tmp\*  for temporary files.
X
SHAR_EOF
chmod 0600 sort.doc || echo "restore of sort.doc fails"
sed 's/^X//' << 'SHAR_EOF' > sortcomp.c &&
X/*****************************************************************************
X *                                                                           *
X *   sortcomp.c  version 1.0 of  1 Januari 1989    (C) L.J.M. de Wit 1989    *
X *                                                                           *
X * This software may be used and distributed freely if not used commercially  *
X * and the originator (me) is mentioned in the source (just leave this 9 line *
X * header intact).                                                           *
X *                                                                           *
X *****************************************************************************
X *
X * sortcomp.c: comparision functions
X *
X * These functions contain the numeric comparision functions c_n and its
X * reverse, c_nr. The other 48 comparision functions are formed by combining
X * the following properties (take note of the Capitals):
X *  a) either Dictionary, Ignore non-printables or Any other        (3)
X *  b) ignore leading Blanks, versus not ignoring them              (2)
X *  c) Fold upper case to lower, versus not folding                 (2)
X *  d) Reverse the result of the comparision, versus not reversing  (2)
X *  e) Unbounded (bounded only by '\0'), versus upper limit bound   (2)
X *
X * Since comparing is done a lot it is important for the compare functions
X * to be fast. As for using the 48 functions as it stands the following
X * arguments are used:
X * One could have combined functions and used flags to discern between options.
X * This would however involve passing more parameters to the function (the
X * flags) and would require (perhaps a lot of) unwanted testing inside the
X * function.
X * Several functions call one of the other functions. This has been done in
X * such a way that duplication of code is mostly prevented, but still this call
X * is limited to only one (both in number and in level).
X * The numerical compare could have been done by a simple atoi(), but the
X * c_n function does the comparision inline, and stops comparing when the
X * bigger one has been found (while atoi needs to do calculation ala
X * 10 * num + digit, and will use all digits).
X */
X
X#include <stdio.h>
```

```
X#include <ctype.h>
X#include "sortmain.h"
X#include "sortcomp.h"
X
X#define SKIPSPACE(bs,bt) while (wspace(*(bs))) (bs)++; while (wspace(*(bt))) (bt)
++
X
X#define NCHARS  256
X#define NORM 0x1
X#define DICT 0x2
X#define isnorm(c)   (stype[(c)]&NORM)
X#define isdict(c)   (stype[(c)]&DICT)
X
X#undef  isdigit
X#define isdigit(c) ((c) >= '0' && (c) <= '9')
X
Xstatic char fold[NCHARS];                       /* For lcase conversion     */
X
Xstatic char stype[NCHARS];                      /* For table lookup         */
X
Xvoid initlookup()                              /* Initialize lookup tables */
X{
X    register int i;
X
X    for (i = 0; i < NCHARS; i++) {
X        fold[i] = isupper(i) ? tolower(i) : i;  /* Init lcase convert array */
X        stype[i] = 0;                          /* Not really needed ...    */
X        if (i >= 040 && i <= 0176) {           /* Ascii non-control        */
X            stype[i] |= NORM;
X        }
X        if (wspace(i) || isalnum(i)) {         /* Whitespace & alphanum    */
X            stype[i] |= DICT;
X        }
X    }
X}
X
Xint c_nr(bs,bt,es,et)                          /* Reversed Numeric         */
Xchar *bs, *bt;
Xchar *es, *et;
X{
X    return c_n(bt,bs,et,es);
X}
X
Xint c_n(bs,bt,es,et)                           /* Numeric compare          */
Xregister char *bs, *bt;
Xchar *es, *et;
X{
X    register int rev, mini;
X
X    SKIPSPACE(bs,bt);
X    if (*bs == '-') {
X        if (*bt == '-') {                      /* Two negatives: reverse   */
X            rev = -1;
```

```
X            bs++;
X            bt++;
X        } else {                                /* bt will be the larger one*/
X            return -1;
X        }
X    } else {
X        if (*bt == '-') {                       /* bs will be the larger one*/
X            return 1;
X        } else {
X            rev = 1;                            /* Both positive            */
X        }
X    }
X
X    while (*bs == *bt && isdigit(*bt)) {        /* Compare & skip eq. digits*/
X        bs++; bt++;
X    }
X
X    if (!isdigit(*bs)) {
X        if (!isdigit(*bt)) {
X            if (*bs == '.' && *bt == '.') {     /* Decimal point            */
X                while (*++bs == *++bt && isdigit(*bt)) ;
X                while (*bs == '0') bs++;        /* Skip possibly trailing   */
X                while (*bt == '0') bt++;        /*     zeroes               */
X                if (!isdigit(*bs)) {
X                    if (!isdigit(*bt)) {
X                        return 0;               /* Tails also equal         */
X                    }
X                    return -rev;                /* bt has more digits       */
X                }
X                if (!isdigit(*bt)) {
X                    return rev;                 /* bs has more digits       */
X                }
X                return (rev == 1)               /* Current digit decides    */
X                       ? (*bs - *bt) : (*bt - *bs);
X            } else {
X                return 0;                       /* Equal numeric strings    */
X            }
X        }
X        return -rev;                            /* bt has more digits       */
X    }
X    if (!isdigit(*bt)) {
X        return rev;                             /* bs has more digits       */
X    }
X    mini = (rev == 1)                           /* Current digit decides:   */
X           ? (*bs - *bt) : (*bt - *bs);         /* Difference into mini     */
X    do {
X        bs++;
X        bt++;
X    } while (isdigit(*bs) && isdigit(*bt));     /* Skip any more digits     */
X
X    if (!isdigit(*bs)) {
X        if (!isdigit(*bt)) {                    /* Strings equally long:    */
X            return mini;                        /* then mini decides        */
```

```
X        }
X        return -rev;                           /* bt has more digits       */
X    }
X    return rev;                                /* bs has more digits       */
X}
X
Xint c_df(bs,bt,es,et)                          /* Dictionary/Fold          */
Xregister char *bs, *bt;                        /* Strings to compare       */
Xregister char *es, *et;
X{
X    --bs;   --bt;   --es;   --et;
X    for ( ; bs < es && bt < et; ) {
X        if (fold[*++bs] != fold[*++bt]) {      /* Increment & compare      */
X            if (isdict(*bs)) {
X                if (isdict(*bt)) {
X                    return fold[*bs] - fold[*bt];
X                } else {
X                    --bs;                      /* Effectively skip *bt     */
X                }
X            } else if (isdict(*bt)) {
X                --bt;                          /* Effectively skip *bs     */
X            }                                  /* (else: skip both)        */
X        }
X    }
X    for ( ; bs < es; ) {                       /* Skip any more            */
X        if (isdict(*++bs)) {                   /* non-dict chars in bs     */
X            --bs;
X            break;
X        }
X    }
X    for ( ; bt < et; ) {                       /* Skip any more            */
X        if (isdict(*++bt)) {                   /* non-dict chars in bt     */
X            --bt;
X            break;
X        }
X    }
X
X    return (et - bt) - (es - bs);
X}
X
Xint c_d(bs,bt,es,et)                           /* Dictionary order         */
Xregister char *bs, *bt;
Xregister char *es, *et;
X{
X    --bs;   --bt;   --es;   --et;
X    for ( ; bs < es && bt < et; ) {
X        if (*++bs != *++bt) {                  /* Increment & compare      */
X            if (isdict(*bs)) {
X                if (isdict(*bt)) {
X                    return fold[*bs] - fold[*bt];
SHAR_EOF
echo "End of part 1, continue with part 2"
echo "2" > s2_seq_.tmp
```

```
exit 0


From -5658187205085233886
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,cee3658c03a7496b,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-02-24 11:58:50 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i018: sort -- UNIX-style sort utility  part02/02
Keywords: shar
Message-ID: <6467@saturn.ucsc.edu>
Date: 24 Feb 89 19:58:50 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1309
Approved: koreth@ssyx.ucsc.edu

Submitted-by: leo@philmds.dts.philips.nl (Leo de Wit)
Posting-number: Volume 2, Issue 18
Archive-name: sort/part02

#!/bin/sh
# this is part 2 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file sortcomp.c continued
#
CurArch=2
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> sortcomp.c
X                } else {
X                    --bs;                       /* Effectively skip *bt     */
X                }
X            } else if (isdict(*bt)) {
X                --bt;                           /* Effectively skip *bs     */
X            }                                   /* (else: skip both)        */
X        }
X    }
X    for ( ; bs < es; ) {                        /* Skip any more            */
X        if (isdict(*++bs)) {                    /* non-dict chars in bs     */
X            --bs;
X            break;
X        }
```

```
X    }
X    for ( ; bt < et; ) {                       /* Skip any more           */
X        if (isdict(*++bt)) {                   /* non-dict chars in bt    */
X            --bt;
X            break;
X        }
X    }
X
X    return (et - bt) - (es - bs);
X}
X
Xint c_if(bs,bt,es,et)                          /* Normalized/Fold         */
Xregister char *bs, *bt;
Xregister char *es, *et;
X{
X    --bs;   --bt;   --es;   --et;
X    for ( ; bs < es && bt < et; ) {
X        if (fold[*++bs] != fold[*++bt]) {      /* Increment & compare     */
X            if (isnorm(*bs)) {
X                if (isnorm(*bt)) {
X                    return fold[*bs] - fold[*bt];
X                } else {
X                    --bs;                      /* Effectively skip *bt    */
X                }
X            } else if (isnorm(*bt)) {
X                --bt;                          /* Effectively skip *bs    */
X            }                                  /* (else: skip both)       */
X        }
X    }
X    for ( ; bs < es; ) {                       /* Skip any more           */
X        if (isnorm(*++bs)) {                   /* non-norm chars in bs    */
X            --bs;
X            break;
X        }
X    }
X    for ( ; bt < et; ) {                       /* Skip any more           */
X        if (isnorm(*++bt)) {                   /* non-norm chars in bt    */
X            --bt;
X            break;
X        }
X    }
X
X    return (et - bt) - (es - bs);
X}
X
Xint c_i(bs,bt,es,et)                           /* Normalized order        */
Xregister char *bs, *bt;
Xregister char *es, *et;
X{
X    --bs;   --bt;   --es;   --et;
X    for ( ; bs < es && bt < et; ) {
X        if (*++bs != *++bt) {                  /* Increment & compare     */
X            if (isnorm(*bs)) {
```

```
X               if (isnorm(*bt)) {
X                   return fold[*bs] - fold[*bt];
X               } else {
X                   --bs;                       /* Effectively skip *bt     */
X               }
X           } else if (isnorm(*bt)) {
X               --bt;                           /* Effectively skip *bs     */
X           }                                   /* (else: skip both)        */
X       }
X   }
X   for ( ; bs < es; ) {                        /* Skip any more            */
X       if (isnorm(*++bs)) {                    /* non-norm chars in bs     */
X           --bs;
X           break;
X       }
X   }
X   for ( ; bt < et; ) {                        /* Skip any more            */
X       if (isnorm(*++bt)) {                    /* non-norm chars in bt     */
X           --bt;
X           break;
X       }
X   }
X
X   return (et - bt) - (es - bs);
X}
X
Xint c_af(bs,bt,es,et)                          /* Fold                     */
Xregister char *bs, *bt;
Xregister char *es, *et;
X{
X   register short minlen;
X
X   minlen = ((es - bs) <= (et - bt)) ? (es - bs) : (et - bt);
X   if (minlen == 0) {
X       return (es - bs) - (et - bt);
X   }
X
X   if (fold[*bs] == fold[*bt]) {
X       for ( ; --minlen != 0 && fold[*++bs] == fold[*++bt]; ) ;
X       if (minlen == 0) {
X           return 0;
X       }
X   }
X
X   return fold[*bs] - fold[*bt];
X}
X
Xint c_dfu(bs,bt)                               /* Dictionary/Fold          */
Xregister char *bs, *bt;
X{
X   register char *fld = fold;
X
X   --bs; --bt;
```

```
X    do {
X        while (fld[*++bs] == fld[*++bt] && *bt) ; /* Skip equal ones      */
X        if (!*bt || !*bs) break;                 /* Stop on '\0'          */
X        if (isdict(*bs)) {
X            if (isdict(*bt)) {
X                break;                           /* Difference found      */
X            } else {
X                --bs;                            /* Effectively skip *bt  */
X            }
X        } else {
X            if (isdict(*bt)) {
X                --bt;                            /* Effectively skip *bs  */
X            }
X        }                                        /* (else: skip both)     */
X    } while (1);
X
X    return fld[*bs] - fld[*bt];
X}
X
Xint c_du(bs,bt)                                  /* Dictionary            */
Xregister char *bs, *bt;
X{
X    --bs; --bt;
X    do {
X        while (*++bs == *++bt && *bt) ;          /* Skip equal ones      */
X        if (!*bt || !*bs) break;                 /* Stop on '\0'          */
X        if (isdict(*bs)) {
X            if (isdict(*bt)) {
X                break;                           /* Difference found      */
X            } else {
X                --bs;                            /* Effectively skip *bt  */
X            }
X        } else {
X            if (isdict(*bt)) {
X                --bt;                            /* Effectively skip *bs  */
X            }
X        }                                        /* (else: skip both)     */
X    } while (1);
X
X    return *bs - *bt;
X}
X
Xint c_ifu(bs,bt)                                 /* Normalized/Fold       */
Xregister char *bs, *bt;
X{
X    register char *fld = fold;
X
X    --bs; --bt;
X    do {
X        while (fld[*++bs] == fld[*++bt] && *bt) ; /* Skip equal ones      */
X        if (!*bt || !*bs) break;                 /* Stop on '\0'          */
X        if (isnorm(*bs)) {
X            if (isnorm(*bt)) {
```

```
X                break;                             /* Difference found         */
X            } else {
X                --bs;                              /* Effectively skip *bt     */
X            }
X        } else {
X            if (isnorm(*bt)) {
X                --bt;                              /* Effectively skip *bs     */
X            }
X        }                                          /* (else: skip both)        */
X    } while (1);
X
X    return fld[*bs] - fld[*bt];
X}
X
Xint c_iu(bs,bt)                                    /* Normalized               */
Xregister char *bs, *bt;
X{
X    --bs; --bt;
X    do {
X        while (*++bs == *++bt && *bt) ;            /* Skip equal ones      */
X        if (!*bt || !*bs) break;                   /* Stop on '\0'             */
X        if (isnorm(*bs)) {
X            if (isnorm(*bt)) {
X                break;                             /* Difference found         */
X            } else {
X                --bs;                              /* Effectively skip *bt     */
X            }
X        } else {
X            if (isnorm(*bt)) {
X                --bt;                              /* Effectively skip *bs     */
X            }
X        }                                          /* (else: skip both)        */
X    } while (1);
X
X    return *bs - *bt;
X}
X
Xint c_afu(bs,bt)                                   /* Fold                     */
Xregister char *bs, *bt;
X{
X    register char *fld = fold;
X
X    if (fld[*bs] == fld[*bt] && *bt) {
X        for ( ; fld[*++bs] == fld[*++bt] && *bt; ) ;
X    }
X
X    return fld[*bs] - fld[*bt];
X}
X
X#ifndef M68000
X/* using asm if 68000 */
X
Xint c_dbfr(bs,bt,es,et)                             /* Dict/Blank/Fold/Reverse  */
```

```
Xregister char *bs, *bt;
Xchar *es, *et;
X{
X    SKIPSPACE(bs,bt);
X    return c_df(bt,bs,et,es);
X}
X
Xint c_dbf(bs,bt,es,et)                         /* Dictionary/Blank/Fold    */
Xregister char *bs, *bt;
Xchar *es, *et;
X{
X    SKIPSPACE(bs,bt);
X    return c_df(bs,bt,es,et);
X}
X
Xint c_dbr(bs,bt,es,et)                         /* Dictionary/Blank/Reverse */
Xregister char *bs, *bt;
Xchar *es, *et;
X{
X    SKIPSPACE(bs,bt);
X    return c_d(bt,bs,et,es);
X}
X
Xint c_db(bs,bt,es,et)                          /* Dictionary/Blank         */
Xregister char *bs, *bt;
Xchar *es, *et;
X{
X    SKIPSPACE(bs,bt);
X    return c_d(bs,bt,es,et);
X}
X
Xint c_dfr(bs,bt,es,et)                         /* Dictionary/Fold/Reverse  */
Xchar *bs, *bt;
Xchar *es, *et;
X{
X    return c_df(bt,bs,et,es);
X}
X
Xint c_dr(bs,bt,es,et)                          /* Dictionary/Reversed      */
Xchar *bs, *bt;
Xchar *es, *et;
X{
X    return c_d(bt,bs,et,es);
X}
X
Xint c_ibfr(bs,bt,es,et)                        /* Norm/Blank/Fold/Reverse  */
Xregister char *bs, *bt;
Xchar *es, *et;
X{
X    SKIPSPACE(bs,bt);
X    return c_if(bt,bs,et,es);
X}
X
```

```
Xint c_ibf(bs,bt,es,et)                         /* Normalized/Blank/Fold    */
Xregister char *bs, *bt;
Xchar *es, *et;
X{
X    SKIPSPACE(bs,bt);
X    return c_if(bs,bt,es,et);
X}
X
Xint c_ibr(bs,bt,es,et)                         /* Normalized/Blank/Reverse */
Xregister char *bs, *bt;
Xchar *es, *et;
X{
X    SKIPSPACE(bs,bt);
X    return c_i(bt,bs,et,es);
X}
X
Xint c_ib(bs,bt,es,et)                          /* Normalized/Blank         */
Xregister char *bs, *bt;
Xchar *es, *et;
X{
X    SKIPSPACE(bs,bt);
X    return c_i(bs,bt,es,et);
X}
X
Xint c_ifr(bs,bt,es,et)                         /* Normalized/Fold/Reverse  */
Xchar *bs, *bt;
Xchar *es, *et;
X{
X    return c_if(bt,bs,et,es);
X}
X
Xint c_ir(bs,bt,es,et)                          /* Normalized/Reversed      */
Xchar *bs, *bt;
Xchar *es, *et;
X{
X    return c_i(bt,bs,et,es);
X}
X
Xint c_abfr(bs,bt,es,et)                        /* Blank/Fold/Reverse       */
Xregister char *bs, *bt;
Xchar *es, *et;
X{
X    SKIPSPACE(bs,bt);
X    return c_af(bt,bs,et,es);
X}
X
Xint c_abf(bs,bt,es,et)                         /* Blank/Fold               */
Xregister char *bs, *bt;
Xchar *es, *et;
X{
X    SKIPSPACE(bs,bt);
X    return c_af(bs,bt,es,et);
X}
```

```
X
Xint c_abr(bs,bt,es,et)                         /* Blank/Reverse            */
Xregister char *bs, *bt;
Xchar *es, *et;
X{
X    SKIPSPACE(bs,bt);
X    return c_a(bt,bs,et,es);
X}
X
Xint c_ab(bs,bt,es,et)                          /* Blank                    */
Xregister char *bs, *bt;
Xchar *es, *et;
X{
X    SKIPSPACE(bs,bt);
X    return c_a(bs,bt,es,et);
X}
X
Xint c_afr(bs,bt,es,et)                         /* Fold/Reverse             */
Xchar *bs, *bt;
Xchar *es, *et;
X{
X    return c_af(bt,bs,et,es);
X}
X
Xint c_ar(bs,bt,es,et)                          /* Reverse                  */
Xchar *bs, *bt;
Xchar *es, *et;
X{
X    return c_a(bt,bs,et,es);
X}
X
X/* Unbounded comparisions start here */
X
Xint c_dbfru(bs,bt)                             /* Dict/Blank/Fold/Reverse  */
Xregister char *bs, *bt;
X{
X    SKIPSPACE(bs,bt);
X    return c_dfu(bt,bs);
X}
X
Xint c_dbfu(bs,bt)                              /* Dictionary/Blank/Fold    */
Xregister char *bs, *bt;
X{
X    SKIPSPACE(bs,bt);
X    return c_dfu(bs,bt);
X}
X
Xint c_dbru(bs,bt)                              /* Dictionary/Blank/Reverse */
Xregister char *bs, *bt;
X{
X    SKIPSPACE(bs,bt);
X    return c_du(bt,bs);
X}
```

```
X
Xint c_dbu(bs,bt)                               /* Dictionary/Blank         */
Xregister char *bs, *bt;
X{
X    SKIPSPACE(bs,bt);
X    return c_du(bs,bt);
X}
X
Xint c_dfru(bs,bt)                              /* Dictionary/Fold/Reverse  */
Xchar *bs, *bt;
X{
X    return c_dfu(bt,bs);
X}
X
Xint c_dru(bs,bt)                               /* Dictionary/Reverse       */
Xchar *bs, *bt;
X{
X    return c_du(bt,bs);
X}
X
Xint c_ibfru(bs,bt)                             /* Norm/Blank/Fold/Reverse  */
Xregister char *bs, *bt;
X{
X    SKIPSPACE(bs,bt);
X    return c_ifu(bt,bs);
X}
X
Xint c_ibfu(bs,bt)                              /* Normalized/Blank/Fold    */
Xregister char *bs, *bt;
X{
X    SKIPSPACE(bs,bt);
X    return c_ifu(bs,bt);
X}
X
Xint c_ibru(bs,bt)                              /* Normalized/Blank/Reverse */
Xregister char *bs, *bt;
X{
X    SKIPSPACE(bs,bt);
X    return c_iu(bt,bs);
X}
X
Xint c_ibu(bs,bt)                               /* Normalized/Blank         */
Xregister char *bs, *bt;
X{
X    SKIPSPACE(bs,bt);
X    return c_iu(bs,bt);
X}
X
Xint c_ifru(bs,bt)                              /* Normalized/Fold/Reverse  */
Xchar *bs, *bt;
X{
X    return c_ifu(bt,bs);
X}
```

```
X
Xint c_iru(bs,bt)                               /* Normalized/Reverse       */
Xchar *bs, *bt;
X{
X    return c_iu(bt,bs);
X}
X
Xint c_abfru(bs,bt)                             /* Blank/Fold/Reverse       */
Xregister char *bs, *bt;
X{
X    SKIPSPACE(bs,bt);
X    return c_afu(bt,bs);
X}
X
Xint c_abfu(bs,bt)                              /* Blank/Fold               */
Xregister char *bs, *bt;
X{
X    SKIPSPACE(bs,bt);
X    return c_afu(bs,bt);
X}
X
Xint c_abru(bs,bt)                              /* Blank/Reverse            */
Xregister char *bs, *bt;
X{
X    SKIPSPACE(bs,bt);
X    return c_au(bt,bs);
X}
X
Xint c_abu(bs,bt)                               /* Blank                    */
Xregister char *bs, *bt;
X{
X    SKIPSPACE(bs,bt);
X    return c_au(bs,bt);
X}
X
Xint c_afru(bs,bt)                              /* Fold/Reverse             */
Xregister char *bs, *bt;
X{
X    register char *fld = fold;
X
X    if (fld[*bs] == fld[*bt] && *bt) {
X        for ( ; fld[*++bs] == fld[*++bt] && *bt; ) ;
X    }
X
X    return fld[*bt] - fld[*bs];
X}
X
Xint c_aru(bs,bt)                               /* Reverse                  */
Xregister char *bs, *bt;
X{
X    if (*bs == *bt && *bt) {
X        for ( ; *++bs == *++bt && *bt; ) ;
X    }
```

```
X
X    return *bt - *bs;
X}
X
Xint c_au(bs,bt)                                /* Ordinary sort           */
Xregister char *bs, *bt;
X{
X    if (*bs == *bt && *bs) {                   /* Compare & skip eq. chars */
X        while (*++bs == *++bt && *bs) ;        /* Compare & skip eq. chars */
X    }
X
X    return *bs - *bt;
X}
X
Xint c_a(bs,bt,es,et)                            /* Ordinary sort           */
Xregister char *bs, *bt;
Xregister char *es, *et;
X{
X    register short minlen;
X
X    minlen = ((es - bs) <= (et - bt)) ? (es - bs) : (et - bt);
X    if (minlen == 0) {
X        return (es - bs) - (et - bt);
X    }
X
X    if (*bs == *bt) {
X        for ( ; --minlen != 0 && *++bs == *++bt; ) ;
X        if (minlen == 0) {
X            return 0;
X        }
X    }
X
X    return *bs - *bt;
X}
X#endif not MC68000
SHAR_EOF
chmod 0600 sortcomp.c || echo "restore of sortcomp.c fails"
sed 's/^X//' << 'SHAR_EOF' > sortcomp.h &&
X/*****************************************************************************
X *                                                                           *
X *   sortcomp.h  version 1.0 of 22 Januari 1989    (C) L.J.M. de Wit 1989    *
X *                                                                           *
X * This software may be used and distributed freely if not used commercially  *
X * and the originator (me) is mentioned in the source (just leave this 9 line *
X * header intact).                                                           *
X *                                                                           *
X *****************************************************************************
X *
X * sortcomp.h: header for sortcomp.c's external definitions
X */
X
Xextern int  c_nr(),     c_n(),                  /* the various compare funcs*/
X           c_dbfr(),   c_dbf(),    c_dbr(),    c_db(),
```

```
X            c_dfr(),    c_df(),     c_dr(),     c_d(),
X            c_ibfr(),   c_ibf(),    c_ibr(),    c_ib(),
X            c_ifr(),    c_if(),     c_ir(),     c_i(),
X            c_abfr(),   c_abf(),    c_abr(),    c_ab(),
X            c_afr(),    c_af(),     c_ar(),     c_a(),
X            c_dbfru(),  c_dbfu(),   c_dbru(),   c_dbu(),
X            c_dfru(),   c_dfu(),    c_dru(),    c_du(),
X            c_ibfru(),  c_ibfu(),   c_ibru(),   c_ibu(),
X            c_ifru(),   c_ifu(),    c_iru(),    c_iu(),
X            c_abfru(),  c_abfu(),   c_abru(),   c_abu(),
X            c_afru(),   c_afu(),    c_aru(),    c_au();
X
Xextern void initlookup();                      /* Initialize lookup tables */
SHAR_EOF
chmod 0600 sortcomp.h || echo "restore of sortcomp.h fails"
sed 's/^X//' << 'SHAR_EOF' > sortfile.c &&
X/*****************************************************************************
X *                                                                           *
X *   sortfile.c  version 1.0 of  1 Januari 1989    (C) L.J.M. de Wit 1989    *
X *                                                                           *
X * This software may be used and distributed freely if not used commercially  *
X * and the originator (me) is mentioned in the source (just leave this 9 line *
X * header intact).                                                           *
X *                                                                           *
X *****************************************************************************
X *
X * sortfile.c: stdio replacement for Atari ST
X *
X * This replacement is added both to speed up the normal stdio offered by
X * Lattice C, and to repair a bug with open() (fopen() ?) that is in the
X * standard library (version 3.03: try opening & closing 20 files, your
X * program bombs when you try to open a new file, even though the old ones
X * have been closed).
X *
X * Note that this package is not complete (no flushbuf, filbuf, printf etc.)
X * but this wasn't the intent anyway; furthermore, the allocation of buffers
X * takes place when a file is opened, rather than when it is first read/written
X * (as in a standard stdio package).
X */
X
X#include <stdio.h>
X#include <osbind.h>
X#include <fcntl.h>
X#include "sortmain.h"
X#include "sortfile.h"
X
X#undef  Fwrite
X#define Fwrite fixwrite                         /* Fix for Fwrite bug       */
X
Xstatic int buflen;                              /* Default buffer length    */
X
Xstatic int lopen();
Xstatic int fixwrite();
```

```
X
X#ifdef BESTIO
X
Xvoid bestio(size)                              /* Initializes stdin/stdout */
Xint size;
X{
X    char *iobuffer[2];                         /* stdin/stdout buffer ptrs */
X
X    buflen = size;
X    iobuffer[0] = (char *)Malloc(buflen);
X    iobuffer[1] = (char *)Malloc(buflen);
X    if (iobuffer[0] == (char *)-1 || iobuffer[1] == (char *)-1) {
X        error("memory allocation failed\n",(char *)0);
X    }
X    stdin->_ptr   = stdin->_base = iobuffer[0];
X    stdin->_rcnt  = buflen;
X    stdin->_flag  = _IOREAD;
X    stdin->_file  = 0;
X    stdin->_size  = buflen;
X
X    stdout->_ptr  = stdout->_base = iobuffer[1];
X    stdout->_wcnt = buflen;
X    stdout->_flag = _IOWRT;
X    stdout->_file = 1;
X    stdout->_size = buflen;
X}
X
Xvoid exit(code)                                 /* Replaces standard exit   */
Xint code;
X{
X    int i;
X
X    for (i = 0; i < _NFILE; i++) {
X        if (_iob[i]._flag != 0) {              /* Stclose any open files   */
X            stclose(_iob+i);
X        }
X    }
X    _exit(code);                               /* and _exit                */
X}
X
X#else NOT BESTIO
X
Xint _fmode = 0x8000;                            /* No translation, for speed*/
X
X#endif BESTIO
X
XFILE *stopen(filename,mode)                     /* Replaces fopen()         */
Xchar *filename, *mode;
X{
X    FILE *fp;
X    int desc;
X    int flag;
X    int i;
```

```
X
X    for (i = 0; i < _NFILE; i++) {             /* Find first free entry    */
X        if (_iob[i]._flag == 0) {
X            break;
X        }
X    }
X    if (i < _NFILE) {
X        fp = _iob + i;                         /* First free               */
X    } else {
X        return (FILE *)0;                      /* All occupied             */
X    }
X
X    if (!strcmp(mode,"w")) {                   /* Mode "w": write          */
X        desc = lopen(filename,O_WRONLY|O_CREAT|O_TRUNC);
X        flag = _IOWRT;
X    } else if (!strcmp(mode,"r")) {            /* Mode "r": read           */
X        desc = lopen(filename,O_RDONLY);
X        flag = _IOREAD;
X    } else {                                   /* Unknown mode             */
X        return (FILE *)0;
X    }
X
X    if (desc < 0) {
X        return (FILE *)0;                      /* Lopen failed             */
X    } else {
X        fp->_rcnt = fp->_wcnt = fp->_size = 0;
X        fp->_ptr  = fp->_base = (char *)0;
X        fp->_file = desc;
X        fp->_flag = flag;
X    }
X
X    return fp;
X}
X
Xint stclose(fp)                                /* Replaces fclose()        */
XFILE *fp;
X{
X    int retco;
X
X    if (fp->_flag & (_IORW|_IOWRT)) {
X        stflush(fp);                           /* Flush an output stream   */
X    }
X    retco = Fclose(fp->_file);                 /* Close file               */
X    if (fp->_base != (char *)0) {
X        retco |= Mfree(fp->_base);              /* Free allocated buffer    */
X    }
X    fp->_base = (char *)0;
X    fp->_flag = 0;                             /* Indicate entry is free   */
X
X    return retco;
X}
X
Xint stflush(fp)
```

```
XFILE *fp;
X{
X    if (fp->_size > fp->_wcnt) {               /* Any data to be flushed ? */
X        Fwrite(fp->_file,fp->_size-fp->_wcnt,fp->_base); /* Write it       */
X        fp->_ptr = fp->_base;                  /* re-init ptr              */
X        fp->_wcnt = fp->_size;                 /* and _wcnt                */
X    }
X}
X
Xvoid stbuffer(fp,buf,size)
XFILE *fp;
Xchar *buf;
Xint size;
X{
X    fp->_ptr = fp->_base = buf;
X    fp->_size = size;
X    fp->_wcnt = size;
X    fp->_rcnt = 0;
X}
X
Xchar *stgets(inbuf,size,fp)                   /* Replaces fgets()         */
Xchar *inbuf;
Xregister int size;
Xregister FILE *fp;
X{
X    register int todo, i, ntoread;
X    register char *s, *t;
X
X    if (fp->_base == (char *)0) {
X        s = (char *)Malloc(buflen);            /* Allocate buffer   */
X        stbuffer(fp,s,buflen);
X    }
X    if (size-- <= 0 || feof(fp)) {             /* One for the '\0'         */
X        return (char *)0;
X    }
X
X    t = inbuf;
X    s = fp->_ptr;
X    ntoread = fp->_rcnt;
X
X    for ( ; ; ) {
X        todo = min(size,ntoread);
X        for (i = 0; i < todo; i++) {
X            if ((*t++ = *s++) == '\n') {       /* Had a newline            */
X                i++;
X                *t = '\0';                     /* 0-terminate line         */
X                fp->_ptr = s;
X                fp->_rcnt = ntoread - i;
X                return inbuf;
X            }
X        }
X        if ((size -= todo) <= 0) {             /* All read                 */
X            break;
```

```
X        }
X        ntoread = Fread(fp->_file,fp->_size,fp->_base); /* Read a buffer    */
X        s = fp->_base;                          /* Re-init ptr             */
X        if (ntoread <= 0) {
X            fp->_flag |= _IOEOF;
X            return (char *)0;                   /* End-of-file */
X        }
X    }
X    fp->_rcnt = ntoread - todo;
X    fp->_ptr = s;
X    *t = '\0';
X    return inbuf;
X}
X
Xvoid stputs(inbuf,fp)                           /* Replaces fputs()        */
Xchar *inbuf;
Xregister FILE *fp;
X{
X    register int i, ntowrite;
X    register char *s, *t;
X
X    if (fp->_base == (char *)0) {
X        s = (char *)Malloc(buflen);             /* Allocate buffer   */
X        stbuffer(fp,s,buflen);
X    }
X    s = inbuf;
X    t = fp->_ptr;
X    ntowrite = fp->_wcnt;
X
X    for ( ; ; ) {
X        for (i = 0; i < ntowrite; i++) {
X            if (*s == '\0') {                   /* Found 0 terminator      */
X                break;
X            }
X            *t++ = *s++;
X        }
X        if (*s == '\0') {
X            fp->_wcnt = ntowrite - i;
X            fp->_ptr = t;
X            return;
X        }
X        Fwrite(fp->_file,t - fp->_base,fp->_base); /* Write a buffer        */
X        ntowrite = fp->_size;
X        t = fp->_base;                          /* Re-init ptr             */
X    }
X}
X
Xstatic int lopen(filename,mode)                 /* Simple open() replacement*/
Xchar *filename;
Xint mode;
X{
X    int fd;
X
```

```
X    if (!(mode & O_TRUNC)) {
X        fd = Fopen(filename,mode & (O_WRONLY|O_RDWR));      /* Open file    */
X    }
X
X    if ((mode & O_TRUNC) || ((fd < 0) && (mode & O_CREAT))) {
X        fd = Fcreate(filename,!(mode & (O_WRONLY|O_RDWR))); /* Create file  */
X    }
X
X    return fd;
X}
X
Xstatic int fixwrite(fd,len,buf)
Xint fd, len;
Xchar *buf;
X{
X    int total, piece = 0;
X    char *top;
X
X    if (fd != 1) {
X        return gemdos(0x40,fd,len,buf);
X    }
X    for (total = 0, top = buf + len; len > 0; buf += piece, len -= piece) {
X        if ((piece = len) > 32767) {
X            piece = 32256;
X        }
X        total += gemdos(0x40,fd,piece,buf);
X    }
X    return total;
X}
SHAR_EOF
chmod 0600 sortfile.c || echo "restore of sortfile.c fails"
sed 's/^X//' << 'SHAR_EOF' > sortfile.h &&
X/*****************************************************************************
X *                                                                           *
X *   sortfile.h     version 1.0 of 22 Januari 1989    (C) L.J.M. de Wit 1989
*
X *                                                                           *
X * This software may be used and distributed freely if not used commercially  *
X * and the originator (me) is mentioned in the source (just leave this 9 line *
X * header intact).                                                           *
X *                                                                           *
X *****************************************************************************
X *
X * sortfile.h: external definitions for the sortfile stdio package.
X *
X * To enable the speeded up std I/O routines, the BESTIO symbol must be
X * #define'd (with -dBESTIO in the LC1 compilation phase when using Lattice C)
X * and sortfile used when linking.
X * To use the standard stdio package, don't define BESTIO and don't link with
X * sortfile.
X */
X
X#ifdef BESTIO
```

```
X
X#define fopen  stopen
X#define fgets  stgets
X#define fputs  stputs
X#define fclose stclose
X#define setbuffer stbuffer
X
Xextern FILE *stopen();
Xextern char *stgets();
Xextern void stputs();
Xextern void stbuffer();
Xextern int  stclose();
Xextern void bestio();
Xextern void exit();
X
X#else not BESTIO
X
X#define bestio(s)
X
X#endif BESTIO
SHAR_EOF
chmod 0600 sortfile.h || echo "restore of sortfile.h fails"
sed 's/^X//' << 'SHAR_EOF' > sortmain.c &&
X/*****************************************************************************
X *                                                                           *
X *   sortmain.c  version 1.0 of 22 Januari 1989    (C) L.J.M. de Wit 1989    *
X *                                                                           *
X * This software may be used and distributed freely if not used commercially  *
X * and the originator (me) is mentioned in the source (just leave this 9 line *
X * header intact).                                                           *
X *                                                                           *
X *****************************************************************************
X *
X * sortmain.c: main module for sort
X */
X
X#include <stdio.h>
X#include <ctype.h>
X#include "sortmain.h"
X#include "sortcomp.h"
X#include "sortfile.h"
X
Xtypedef struct key {                            /* Sort key structure       */
X    int (*cmp)();                              /* Comparision routine      */
X    int bfld;                                  /* Begin field no.          */
X    int bchar;                                 /* Begin char no.           */
X    int echar;                                 /* End char no.             */
X    int flds;                                  /* # of fields (efld-bfld)  */
X    int bis0;                                  /* bfld == bchar == 0       */
X    int eis0;                                  /* efld == echar == 0       */
X    struct key *next;                          /* Next sort key ptr.       */
X} key;
X
```

```
Xint options = 0;                               /* Global sort type flag    */
Xlong _mneed = 40000;                           /* Memory occupied by prog  */
X
Xstatic char sepchar = ' ';                     /* Field separator char     */
Xstatic key firstkey;                           /* First key used           */
X
Xstatic int compar();                           /* Standard compare routine */
Xstatic int compb0(), compe0(), compbe();       /* Special cases of compar()*/
Xstatic int bdfinr();                           /* Get 'bdfinr' options     */
Xstatic char *(*fieldfunc)();
Xstatic char *getsepfld(), *getspafld();        /* Search field + offset    */
Xstatic int (*getcmp())();
Xextern void allsort();
X
Xmain(argc,argv)
Xint argc;
Xchar **argv;
X{
X    int i, qualif;
X    char *s;
X    char *outfile = (char *)0;                 /* Result file (stdout)     */
X    key *keyp, *keyn;
X    int (*compfunc)();                         /* Routine used to compare  */
X    int nkeys = 0;                             /* No. of keys to be used   */
X
X    settemp("\\tmp");                          /* Set default temp dir     */
X
X    initlookup();                              /* Initialize lookup arrays */
X
X    for (i = 1; i < argc && (*argv[i] == '-' ||
X                        *argv[i] == '+'); i++) { /* Handle commandline flags */
X        s = argv[i];
X        if (*s++ == '-') {
X            if (*s == '\0') {                  /* '-' denotes std input    */
X                break;
X            }
X            for ( ; *s != '\0'; ) {
X                if (qualif = bdfinr(*s)) {     /* *s in (b,d,f,i,n,r)      */
X                    options |= qualif;         /* Add it to options        */
X                    s++;
X                } else {
X                    switch (*s++) {
X                    case 't':                  /* Field separator char     */
X                        sepchar = *s++;
X                        break;
X                    case 'c':                  /* Don't sort, only check   */
X                        options |= CHECKONLY;
X                        break;
X                    case 'm':                  /* Merge the sorted input   */
X                        options |= MERGEONLY;
X                        break;
X                    case 'o':                  /* Output file              */
X                        options |= OUTFILE;
```

```
X                       s += strlen(s);
X                       outfile = argv[++i];
X                       break;
X                   case 'T':                   /* Directory for temp files */
X                       options |= SEPARATOR;
X                       s += strlen(s);
X                       settemp(argv[++i]);
X                       break;
X                   case 'u':                   /* Remove duplicate lines   */
X                       options |= UNIQUE;
X                       break;
X                   default:                    /* Invalid option           */
X                       *s = '\0';              /* Null terminate           */
X                       error("invalid option %s\n",--s);
X                       break;
X                   }
X               }
X           }
X       } else {                                /* '+' sort key flag        */
X           int keyflags = 0, efld = 0;
X
X           if (nkeys++ == 0) {
X               keyp = &firstkey;
X           } else {
X               keyp = (key *)malloc(sizeof(key));/* Create the (next) key  */
X               if (keyp == (key *)0) {
X                   error("memory allocation failed\n",(char *)0);
X               }
X               for (keyn = &firstkey; keyn->next != (key *)0;
X                                                   keyn = keyn->next) ;
X               keyn->next = keyp;
X           }
X           keyp->next = (key *)0;
X           keyp->bfld = atoi(s);               /* Get key's start field no.*/
X           for ( ; isdigit(*s); s++) ;
X           if (*s == '.') {
X               keyp->bchar = atoi(++s);        /* Get char offset for field*/
X               for ( ; isdigit(*s); s++) ;
X           } else {
X               keyp->bchar = 0;                /* 0 is default             */
X           }
X           for ( ; *s != '\0'; s++) {          /* 'bdfinr' flags for key   */
X               if (qualif = bdfinr(*s)) {
X                   keyflags |= qualif;         /* Add flag                 */
X               } else {
X                   s[1] = '\0';
X                   error("invalid option %s\n",s);
X               }
X           }
X           if (i + 1 < argc && argv[i+1][0] == '-'  /* End field specified */
X               && (argv[i+1][1] == '.' || isdigit(argv[i+1][1]))) {
X               s = argv[++i] + 1;
X               efld = atoi(s);                 /* Get key's end field no.  */
```

```
X               for ( ; isdigit(*s); s++) ;
X               if (*s == '.') {
X                   keyp->echar = atoi(++s);    /* Get char offset for field*/
X                   for ( ; isdigit(*s); s++) ;
X               } else {
X                   keyp->echar = 0;            /* 0 is default             */
X               }
X               for ( ; *s != '\0'; s++) {      /* 'bdfinr' flags for key   */
X                   if (qualif = bdfinr(*s)) {
X                       keyflags |= qualif;  /* add flag                */
X                   } else {
X                       s[1] = '\0';
X                       error("invalid option %s\n",s);
X                   }
X               }
X           } else {
X               efld = keyp->echar = 0;
X           }
X           if (keyflags == 0) {                /* If no field flags        */
X              keyflags = options;              /* Take global ones         */
X           }
X           keyp->flds = efld - keyp->bfld;
X           keyp->bis0 = keyp->bfld == 0 && keyp->bchar == 0;
X           keyp->eis0 = efld == 0 && keyp->echar == 0;
X           keyp->cmp = getcmp(keyflags,keyp->eis0);
X       }
X   }
X
X   bestio(2048);                               /* Possibly initiate bestio */
X
X   if (nkeys == 0) {                          /* No key specified         */
X       nkeys++;
X       firstkey.next = (key *)0;
X       firstkey.cmp = getcmp(options,1);
X       firstkey.bfld = 0;
X       firstkey.bchar = 0;
X       firstkey.echar = 0;
X       firstkey.flds = 0;
X       firstkey.bis0 = 1;
X       firstkey.eis0 = 1;
X   }
X
X   fieldfunc = (options & SEPARATOR) ? getsepfld : getspafld;
X
X   if (nkeys == 1) {                           /* One key only             */
X       if (firstkey.bis0) {
X           if (firstkey.eis0) {
X               compfunc = firstkey.cmp;
X           } else {
X               compfunc = compb0;
X           }
X       } else {
X           if (firstkey.eis0) {
```

```
X                compfunc = compe0;
X            } else {
X                compfunc = compbe;
X            }
X        }
X    } else {
X        compfunc = compar;
X    }
X    allsort(argv+i,outfile, compfunc);
X}
X
Xstatic int (*getcmp(flags, unbounded))()
Xint flags, unbounded;
X{
X    /* DICTORDER, IGNBLANK, FOLDUPPER, REVERSE, unbounded */
X    static int (*cmparr[])() = {                /* All 48 other cases       */
X        c_a,    c_au,   c_ar,   c_aru,
X        c_af,   c_afu,  c_afr,  c_afru,
X        c_ab,   c_abu,  c_abr,  c_abru,
X        c_abf,  c_abfu, c_abfr, c_abfru,
X        c_d,    c_du,   c_dr,   c_dru,
X        c_df,   c_dfu,  c_dfr,  c_dfru,
X        c_db,   c_dbu,  c_dbr,  c_dbru,
X        c_dbf,  c_dbfu, c_dbfr, c_dbfru,
X        c_i,    c_iu,   c_ir,   c_iru,
X        c_if,   c_ifu,  c_ifr,  c_ifru,
X        c_ib,   c_ibu,  c_ibr,  c_ibru,
X        c_ibf,  c_ibfu, c_ibfr, c_ibfru,
X    };
X
X    if (flags & NUMERIC) {                      /* Numeric sort             */
X        return (flags & REVERSE) ? c_nr : c_n;
X    }
X
X    return cmparr[unbounded +
X                  ((flags & REVERSE) ? 2 : 0) +
X                  ((flags & FOLDUPPER) ? 4 : 0) +
X                  ((flags & IGNBLANK) ? 8 : 0) +
X                  ((flags & DICTORDER) ? 16 :
X                          (flags & IGNONPRINT) ? 32 : 0)];
X}
X
Xstatic int bdfinr(c)                            /* Get 'bdfinr' options     */
Xchar c;
X{
X    switch (c) {
X    case 'b':                                  /* Ignore leading blanks    */
X        return IGNBLANK;
X    case 'd':                                  /* Dictionary order         */
X        return DICTORDER;
X    case 'f':                                  /* Fold ucase to lcase      */
X        return FOLDUPPER;
X    case 'i':                                  /* Ignore nonprinting chars */
```

```
X        return IGNONPRINT;
X    case 'n':                                  /* Numeric sort            */
X        return NUMERIC;
X    case 'r':                                  /* Reverse comparision     */
X        return REVERSE;
X    default:                                   /* Other flags             */
X        return 0;
X    }
X}
X
Xstatic int compar(s,t)                         /* Gen.compare for all keys */
Xchar *s, *t;                                   /* Strings to be compared   */
X{
X    register key *keyp = &firstkey;            /* Start with key firstkey  */
X    register int result;                       /* Result of comparision    */
X    register char *bs, *bt;                    /* Begin pos within strings */
X
X    do {
X        if (keyp->bis0) {                      /* Avoid function call    */
X            bs = s;
X            bt = t;
X        } else {
X            bs = (*fieldfunc)(s,keyp->bfld,keyp->bchar);
X            bt = (*fieldfunc)(t,keyp->bfld,keyp->bchar);
X        }
X        if (keyp->eis0) {                      /* Unbounded end;         */
X            result = (*keyp->cmp)(bs,bt);      /* Avoid end calculation    */
X        } else {
X            result = (*keyp->cmp)(bs,bt,
X                    (*bs == '\0') ? bs :
X                        (*fieldfunc)(bs-keyp->bchar,keyp->flds,keyp->echar),
X                    (*bt == '\0') ? bt :
X                        (*fieldfunc)(bt-keyp->bchar,keyp->flds,keyp->echar));
X        }
X    } while (result == 0                       /* Strings compare equal    */
X        && (keyp = keyp->next) != (key *)0) ;  /* and more keys to consider*/
X
X    return result;
X}
X
Xstatic int compb0(bs,bt)                       /* Compare for 1 bis0 key   */
Xregister char *bs, *bt;                        /* Strings to be compared   */
X{
X    register key *keyp = &firstkey;            /* For efficiency only      */
X
X    return (*keyp->cmp)(bs,bt,
X                (*fieldfunc)(bs,keyp->flds,keyp->echar),
X                (*fieldfunc)(bt,keyp->flds,keyp->echar));
X}
X
Xstatic int compe0(bs,bt)                       /* Compare for 1 eis0 key   */
Xregister char *bs, *bt;                        /* Strings to be compared   */
X{
```

```
X    register key *keyp = &firstkey;             /* For efficiency only      */
X
X    return (*keyp->cmp)((*fieldfunc)(bs,keyp->bfld,keyp->bchar),
X                        (*fieldfunc)(bt,keyp->bfld,keyp->bchar));
X}
X
Xstatic int compbe(bs,bt)                        /* Compare for 1 bounded key*/
Xregister char *bs, *bt;                          /* Strings to be compared   */
X{
X    register key *keyp = &firstkey;             /* For efficiency only      */
X
X    bs = (*fieldfunc)(bs,keyp->bfld,keyp->bchar);
X    bt = (*fieldfunc)(bt,keyp->bfld,keyp->bchar);
X    return (*keyp->cmp)(bs,bt,
X                (*bs == '\0') ? bs :
X                    (*fieldfunc)(bs-keyp->bchar,keyp->flds,keyp->echar),
X                (*bt == '\0') ? bt :
X                    (*fieldfunc)(bt-keyp->bchar,keyp->flds,keyp->echar));
X}
X
Xstatic char *getsepfld(s,field,ch)               /* Get position of field    */
Xregister char *s;                                /* String to consider       */
Xregister int field, ch;                          /* Field no. and offset cnt */
X{                                               /* Separator was specified  */
X    register char sc = sepchar;
X
X    if (field > 0) {
X        for ( ; *s != '\0'; ) {
X            if (*s++ == sc && --field == 0) {   /* Count 'field' separators */
X                break;
X            }
X        }
X    }
X    for ( ; ch-- != 0; ) {                      /* Go ch characters further */
X        if (*s++ == '\0') {
X            return --s;                         /* but not past '\0'        */
X        }
X    }
X    return s;
X}
X
Xstatic char *getspafld(s,field,ch)               /* Get position of field    */
Xregister char *s;                                /* String to consider       */
Xregister int field, ch;                          /* Field no. and offset cnt */
X{                                               /* No separator; is blanks  */
X    if (field > 0) {
X        for ( ; *s != '\0'; s++) {
X            if (wspace(*s)) {
X                do {
X                    s++;
X                } while (wspace(*s)) ;
X                if (--field == 0) {
X                    break;
```

```
X                }
X            }
X        }
X    }
X    for ( ; ch-- != 0; ) {                     /* Go ch characters further */
X        if (*s++ == '\0') {
X            return --s;                        /* But not past '\0'        */
X        }
X    }
X    return s;
X}
X
Xvoid error(format,arg)                         /* Display a formatted      */
Xchar *format, *arg;                            /* error message and exit   */
X{
X    fprintf(stderr,"sort: ");
X    fprintf(stderr,format,arg);
X    exit(1);
X}
SHAR_EOF
chmod 0600 sortmain.c || echo "restore of sortmain.c fails"
sed 's/^X//' << 'SHAR_EOF' > sortmain.h &&
X/*****************************************************************************
X *                                                                           *
X *   sortmain.h     version 1.0 of 22 Januari 1989    (C) L.J.M. de Wit 1989  *
X *                                                                           *
X * This software may be used and distributed freely if not used commercially  *
X * and the originator (me) is mentioned in the source (just leave this 9 line *
X * header intact).                                                           *
X *                                                                           *
X *****************************************************************************
X *
X * sortmain.h: external definitions & symbols for sortmain
X */
X
X#define IGNBLANK   0x001                        /* Ignore leading blanks    */
X#define DICTORDER  0x002                        /* Dictionary order         */
X#define FOLDUPPER  0x004                        /* Fold ucase to lcase      */
X#define IGNONPRINT 0x008                        /* Ignore non-printing chars*/
X#define NUMERIC    0x010                        /* Interpret as numeric     */
X#define REVERSE    0x020                        /* Reverse result           */
X
X#define CHECKONLY  0x040                        /* Don't sort; check only   */
X#define MERGEONLY  0x080                        /* Merge; input is sorted   */
X#define OUTFILE    0x100                        /* Output was specified     */
X#define UNIQUE     0x200                        /* Remove identical lines   */
X#define SEPARATOR  0x400                        /* Separator was specified  */
X
X#define wspace(c) ((c) == ' ' || (c) == '\t' || (c) == '\n')
X
Xextern int options;                             /* Global sort flags        */
X
Xextern void error();                            /* Display error msg & exit */
```

```
SHAR_EOF
chmod 0600 sortmain.h || echo "restore of sortmain.h fails"
rm -f s2_seq_.tmp
echo "You have unpacked the last part"
exit 0


From -7651608662824883471
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,fe75a6619a681bca,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-02 14:06:26 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!csd4.milw.wisc.edu!bionet!agate!
saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i019: uw -- Unix Windows windowing terminal software  part01/05
Keywords: shar
Message-ID: <6551@saturn.ucsc.edu>
Date: 2 Mar 89 22:06:26 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1217
Approved: koreth@ssyx.ucsc.edu

Submitted-by: sun.com!laidbak!katzung (Brian Katzung)
Posting-number: Volume 2, Issue 19
Archive-name: uw/part01

            **** Client-Side Information ****

                UW (Unix (tm) Windows) ST
          A Windowing Terminal Emulator for the Atari ST

This program was written by Hans-Martin Mosner at the University of
Dortmund in West Germany.  It has since been enhanced by Brad Bosch
and Brian Katzung at Lachman Associates, Inc., in Naperville, Illinois.
We have added quite a lot, including

     o keyboard and mouse locking,
     o simple text capture,
     o kermit file transfer in a uw window,
     o subprocess execution,
     o function-key support,
     o RS232 configuration,
     o additional fonts, and
     o printer support.

Limitations:
     o only supports monochrome systems
     o only supports adm31 emulation
       (but the emulation has been "enhanced"--see README)
     o only supports the minimum UW protocol
```

We have spent a great deal of effort trying to work out or around bugs
in GEM, HMM's code, and our code, but I'm sure some have succeeded in
eluding us even though this code has been in use at Lachman Associates
for over a year.

**** Server Information ****

The latest Unix(tm)-side server that we know of is version 4.1.2.
The author and contact person is:

     John D. Bruner
     ARPANET/MILNET: jdb@s1-c.ARPA
     UUCP: {seismo,ut-sally,dual,lll-crg}!mordor!jdb

-- Brian Katzung ...!laidbak!katzung
   Brad Bosch   ...!laidbak!brad

   Lachman Associates, Inc.
   1901 North Naper Blvd.
   Naperville, IL 60540
   (312) 505-9100

[This contains the source code, as well as a uuencoded arcfile with
 resources, fonts, and so on. The binaries are being posted to the
 binaries group. -sg]

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
# This is part 1 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
#
#    Run the following text with /bin/sh to create:
#      BINARC.UUE
#      BLDMFORM.C
#      KERMIT.C
#      LASTCHG
#      LCKBLK1.ICN
#      MAKE.BAT
#      MAKEFILE
#      MFORM.C
#      README
#      RESETAUX.S
#      RMBBLK1.ICN
#      STARTGEM.S
#      UW.H
#      VERSION
#      WIN-.CNF
#      WIN.CNF
#      WIND.H
#      WINDEFS.H
#      WINHASH.C
#      WINIO.C
```

```
#      WINMAIN.C
#      WINPROC.C
#      WINPROTO.C
#      WINSUBR.C
#
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
sed 's/^X//' << 'SHAR_EOF' > BINARC.UUE &&
Xbegin 600 bin.arc
XM&@A724Y$+E)30P`"&H@`8!@``#D2V[8:K,8O```,``)8,#'!R<""!0&00.CD
XMA:,`#P`X`*`!`"0$+XR!(%)FSAH0`$`82<.F#$@03^#02?/&S9R3*<O(";.R
XMY<N05]*X(?/FSDDC+>F<A")')QV9)T$$$?.FCE`Z:-*\G$.GCADS(%M`HA!R
XM8\>/0<:,Z3CGC9P\(&*`2,K1HU*Q9,VBE;&V:UNP<.>4/0MB1MVU=]^.U2L7
XM!(V_7MV&';P7;0W$@1?'Y6L#<90Z:816N0("CIPW=-X`T"JA*Y,W8<B`&-/2
XM3)HS7:>$L6.2M1O7L-<.":.RCAR3D`ITO9PYZ>DQ']>4R<,TC!S5!=9T+8*G
XMS!BG)H>T(1.[C%`H--&`0#&&3(JN2V2VR1P;34^1OTTV*=.F:\Z=[XVP:5IT
XM#IK8;*1!ADPO0;)`5U1D5A((0C@'`B0*3$=&<4;4X<880J67QQQ=!5''A&(L
XM*$09;+#QH`%=62%5&B&:-&*))R+X!AP@M,0@B29"<D"'>LDDE!143/$"$51(
XML58!W*TE14?>@0"DD$0:"<D`!?RG6VNO^6:2%%/(,`-=4W:UY!Q-<NDEF#MJ
XMQ=5:3]`F!U5%*0="$V\,*&89;H31!G`'%"!'>T6Y\9&.!7"X%A("FD3%C`\6
XM4``=[;TG!&BAM7&D&%T]`1524Z2A!W!4BK86G;0]F$`!]>FV'YD/(E#`&-V5
XMA"$(2;@!AU,/$A"K=72L`!Y5)A6`1U=#E.3@%&/$Y\:=<WA:1E=.E.'3?3SY
XM-&4!=T`K;5]TX0$"#C#DJJU/,ASF+;A'/KM6M#[%T.VWX4*BZ[K;X@""MS8<
XM%MRX*&TJ1ZX%++O6<6&8.,2,>71EK[<Q5#;'AB"8$92X:U7F[0UK/?R2Q&X(
XM)6]7&#/L5T]N1#PQ"`6PT97%:85;L%`<"X5R&%V9"P+&(-#A!EHQ'YE&5UC<
XM"T(6:X$F7L\?#_S&&0Q2^H:EU\)AVM(@+$KC15T1910(U+X7YEI3A';UJ7$`
XMT$0=;*S40M<^4:&>3@6#4$0;:-/$TK)")(Q$&"ZUT(1S*Y5,YQQNR`1`%6ZD
XMX6:S=*#U!E9$F$4'W3L!\(1T=/Y&ZVVB47>42XJ3B-;>R)6A6MX`"%$4WR`L
XM09,>%L*F>FI-SS'&?TR$<7L;K`=!V!AIT-01"YN/X0(`7T!2`?+*`X!%\^R"
XMX(2>9>A@^1(`#!&$$T,4P00`4Y#(:\YHF&2&A1C>#8)R:(4&@NG%U=&L&[!-
XMN2D(8C@5FDLU_BL&6K^!`QMTIQ.FW>]\%ZK)LJ:4ISVY8"U4*%_)X/>4\D4,
XM?0I<WW+P5Z<\'&]*T<*#4-@'`**4P0X:3-@4W.,3II`A83PI''!$X`41@&!/
XM?'N);>CP&3;L8"U>\%8/5L-#-K0`#\3S@A=`,$0O6&0`&@)!`ZLG$N1]P8K.
XMPT+J.FB]Y*&`>5]\WA>-H!80M$`D,7"9YBSDGS28X2BJ06#Z;,2^.7QP`&14
XMRQF-(`,UFD1WC#$=".Z0&?'LL(=WY*,>13*#<+7QC2#@FVJ,,(-%&H$&+DN;
XM3/)TE#L6H6-(F=0+`?#)H_SK?FZH0QO$@)3'@>`S=WB))%?S!C:HTB4`D-@I
XM+5@XGQ`2/W<XGA1Z,@?K'<R6;7"),?DV%I4]`7M?N&(TL\B\`533>6&*H%1`
XM0*8XU`%/8ZD15FRGK)>(H2AE,`,;8-B19+&H(]Q\FDE<>;^2Z(5\K'N#&.S`
XM$OF!Q`QEH(F67D+/J+S$,TN;21L>R"!8SH]ICS/;$Z10A.L!X'Y[2J4L=_(^
XM/,!!)FG(*!WL9B/GF"0J9T##.@'P&]9D=$!D8"CXY'E#/-7A@@F\VTO0,)L_
XMEBP,OPL>'`$`@)>U@'WXT\F$Z,?0)&`E#"#(#'U`P-.IU.%VD00`VW3(.E9R
XM4WP8$N3_I*A*5OZ+D%`!B1B46D#B%6>;;@"-('5"/I-D]*9H%4\!CG>_EZ5P
XM/`50"YED18>7_+)::2%>`60`@,&.[[#OH8MWQI`"AFXUDK\IJEY>4SC5'#`-
XM;Q+*;,)`DC"TB*RK-!Q5`%?`018R+8.48%W18M(:?;2SQQN1+DU2!Y]L<[5R
XM@&.-2G8_]U"%FXEB`0!:LL[AF@2ROI6E'4@[P!8Q-"4*Y%"MR+?-Q=V->`'%
XM*G3?1[<!'F6C%^6I:#4'U&:=H;.1%,J,\)2SD-:FJ\^*(),Z^KEFV02S\W1*
XM@.!+UTT5E0QMJ&3.WI8G$^4UGGOB+Y[\ZY+C/5.B%`7`[?AV!D'JDKN&5>I[
XMQNK8]-&OKNG](T(](U2[VC2J1VG#'(BWJ9*-854%3'%4$[>2N$%7F$4(`A&:
XM4(2(D<0D4*'):JCGVK1"99L(/8-"+8R])DRTHL5*`W):>[^BI%0H;6@*J_)'
XMA_WM&,53NBO^'#0@,H%N)9]Z"49?+`;GW#'+R(GD&4A+W#>\LB.A^<T=.T6_
XM!=U8RVO@L@6]C`8PB]DD9-[?$W72K`%%TK7`_"IA93E(L\31+$^\CAQ^TS$0
```

```
XM#+@,Q)LE&69RAC.TML2%S>J4?N/JEMR1-7#(0UM!P),ZG/;06SXQK.5<OB<2
XM^@R&#E"P#;CHUS3ZAH_&G_Y:\L228AJQ3'F.3(@WAP`-2`[$,TM]Z;"@.O]I
XM2L.NZ[7?\V&=W(H.#(V<KTL2:F4G6MAX(L-&T2)J4I,;+;`VW1./(L([7MC*
XM&=8O@W2WACG`07>H]DI)CD(\`J2:HP-(X4MXES"ORD^0[@MSJ9S3%([.DH)-
XMCLJR)E0PJKF&1/K&IU#\\Y[[\6$U/)U)6.5P/-V94I::H[2/!!F&-R+%YL=3
XM>!%L]]<;!_1-ZGXY&V)*+Y]L6`XM`$`,(8T6SY!%T299^AC^*K$2]42L"5.X
XM:PS-FZIH;D89W*;;X?LXK/Q/P\8*%+/-=V0I4H^AVGS)-@/:K)7>;TES3R&5
XM,5Q1H/PKS$'GG!QXI\!PG]+/^API76F&:\=A93/$VZ'/&TJ[2=E./.)67?"6
XMY3HZP.[$2CY@+??S2Z:E9M5DL5X,<@`#M3A!<N*9WD?_-2FTG=`YW,G=[GH7
XM5.%Q2/@RX6>)(IX$)E0,!C7`Y(,&@((J3"$(YT%!)660`A#4`/LMR$$:89"]
XM[77O>Q<VTY=`,(0G.,$(23A"%:00!"HDP?ZI$P8?XB3"DSJ9,0<O,`0Y!P"[
XMUT?L!P(Y8`,P\(`T`"X/.`,2^(#E,H$AT4=?$P,6&!+NPH'7]X!]08(@X(!=
XM$0/GMX(S0`,KF$8KV$<A,27@`@`U```V```W```X4$+.D1D)\WLR5`2TL2Q/
XM0`;<$3:,(@0'"`#Z\1X'TS$]Q(`,Z`(YR%A$^"Q88'\OT(5&8`0`\"0)&"0`
XM(`,UL(,R@#T-@SVU\FX,8A4`]2^=\BDE9"RLTC@?=6D/IQ=WX&FQI67B03?'
XMQ4H`\'&>Y6<6LA]Y=C_LTQS/$4D<%68?EW1YL(?WTX=S\(>12`>$%$YD\G`S
XM(5QW1U%4L'].((F?9G9W@'9.4@2G*`5.``CE0R8`0&D\=%79Y5HP\AN\0U<6
XMLA(F<C^>*!J:R(DQ1WFWPU!*(#]"X77W9(JH>%&?B&1^-@;,1"(@9ELR05)N
XMP%`FI!<`$`1&MTLF<8R>!G2U818M]6\`@">F9#HT9D&/B!J1.$N4R"J$!".&
XMR(C*H1K!2!+JAHZ1.'B@]!LQ14I,YSX?IVX#=%P$F9"_=Q0ZL!:.MXW^F#-^
XMYE5D(FH1EP=-T5CN@3:JX29NU#[JU1F_L5DGUF5,`BF19B.FM!Z<!$_[X6HE
XM0R:;91/'4P184`1#4`54@&7NYSU@=)1B!(0*Y1T$8CW481W807]/PSN5$SX<
XM-1+TMB1C$3HB<60<LAL70B+@`U89\C;?X1S4\W,`D`2\TV%S4B?/XCO`DP9O
XM:2=3Z"9"@94F,3U[PB'R4@!50%31(9B!``#6A"(`$`&`0%3_\`\`8`&/*9BK
XM()C1()C28)@`@)B*R9B.:0(!(9C3()C\()C]@)F:N9@`T)@`<!X!()C^0%0&
XMX)F9>0*F252;F9J.^46MF9E)`)MEDYE_<IBVB9JJJ9NP^0>P*0BP60BUF9C$
XMZ9C69$V9Z0BP>0NP"0[->9NJ>1[G`0`'D`!$=0#LYYTQD)W/"0"TN9L'P%C>
XM>07AJ47"Z9R<"0`E\)G>F07AJ9S>.0CF.9^L&9[A$)[D$)[GT)^XB9G2>0#J
XM$)[N$)[R8*"JJ2NZ`@`+T`5$M0"$<*',&9_:Z9CI>:&&<*&J<*&3R:'G20+V
XMN0"L<*'$<*'%`*&.B0$I:@P7V@X7VJ`F.I\?2J'O0%0,4!H`P``3`*,`,`+V
XMR0!<$:0YX*/6DZ,'6I^[R0`[X*-#X*-$0*0(<*05%:1JX*/2X:2J":4^JC)!
XM.@D^2@E$NJ,,4`D^Z@L^^@M$^I]!"@P^:@X^6J!@VIE'B@Y$U0`3`0`-$!%Y
XM"@">N9L-``%]"IF`>@%$BB*[*0&!"0`2X`=$)0&G$*>KJ9JM&2$98!$-``4`
XMT*D1(@``0*I<`0D&$`$680!0(`"MV9K6!)Z="@D)0*HV.`"M20#_D#06<0`H
XM&AP&@*L`4`"[.B^]N@*9^43!VIH&4*PAT:N,Q0!/E`#">@#."A(ZP@,`L#Q3
XMP@#"B@#7VJN]&0)/A`#"F@#AJB-2L)K**JP"D*X'@)\O,*W".B4'TINSJ@#"
XM"JI#RA72VJ\60:RJ6JI/Q!59V@#.R@&]"BO!P16MZ0`)6ZY/E"U3TIH."P`/
XM$+$ZP@A/9*J?"0$:>P"A\$36=+$1$+*V\$2Z<K$2$+++\$3"<;$3$++8.24H
XM<K$+$++U\$0[<K$@L3P?``"G*K!$E:534AH'8@$::P`!$"'!41JM>0$:.P!9
XM&A`5,276!+4`@`%+&P!9=RV2^ID9T+53.B6ZHK4:T+6]:;-AVYH;T+5:5+%M
XM"P`<T+5@T+%SVP%=ZY<#`)Y:ZP%=6YA3DJ5:^P%=&PD\.[<4T+6:\$01HK7'
XM0ZY9-[3_,+`3,25<T0`S-+43&@`C"K87.P*<2U0!P`L!>[$D,+H!49ER>[$E
XMT+4%.KA"^YDFH+H!D`]X>[$GH+H"<+D#4+*?B0)+*P!&:K:SVYHI,+P_."7@
XM>;$J,+Q-X+C'"P`K,+R88J_3RP+#2[$#(*T7VP)36[4"P+%3TK.?"0+#.[)L
XM>['6,Z]+2KD#RZA3HJH'`@/#.PL!JZJM&0/#^Z)RJ[]G.+S7@+<`/`/#FPXD
XMFYB?20/#NP\JJ\"M60-+.P":"[8`;`,3'+1L"\`W,,'E6;X0[(,3?*6R"\`Y
XM,,%6,*TA[`(3_"Q3$B$`+`0`,*5!,+O!4;E$5;P#H*H[P@,3O$`%`,`],,&+
XM^;^?Z0,3+`D$_)D_,,&DD,``#`03G++&"\!!,,'(`+,AK`,3K`W*&L*]6:5(
XM8,-$2[U/M#PH0@03'+L%L#RM60037)IRZ\90N+0$,*130JIT?`1V?!Y8NZV?
XMB01V_`,/3,="8,>!";9T'"%*8#9DC,-6.+\`H"M+8,?2$1P`S`1VW`=/U)H`
XMW`1V_`A+W)I)$+'82@`$H+Z_"\&:"@`1<JH#@*@"H*JH:J2J^JX?H["H2@#X
XM2ZJM&;2P:L>AB:C6Y,N3;,?6``#@*1Q:2ZRYO*H$@,`[@B*0R:S.6A$[\I>"
XMB;$`D*6M::T?4Q'"\9?`&Q$1TIK@^LRH6@":NR-92L?HJLX&4`!&>B`'HKF;
```

```
XMNK2!%:2NW,^[:<H600`%H*TH(JTR&@"MO+)/),NT;`!&BB*XK"L=T,T!G1#2
XMVIIT/`#Z?`D$VYK@F:O7[)T!70`C2ZJZXLV,&<[T.=)/W*DG'1"MC*BPS-"K
XMRE@1$=&`C*H3T*76U)IBJM$?PZT&,`$A6LP`8)RZ&M2K.@&B@*#H^9G.K"OB
XM'-`&\$76I+DHW:PJ/<X$8`#E.1%8_9G@+-7#2M76TZF:>\[='-)<;0!7ZIEA
XM[='.*M03X*;6I"L?J@!+:P`<$JL4^IDY^S$3;;1W[,43`9[&S`![C0B8>2#"
XMT9H(*]@4+2\38`\4?2#&";%*C:H44!K6)!R%BK%SO:H4<#S"B:*M";*;;0`4
XMT,C6M"/&F9HJ79_R<@`9VJG@><ZMG*4S7:H-K5Q9BM.Z;``'(,H]K1"?"=2Z
XM,MP'0`N82:I#"M+R?`!9#)ZZ\MC#NK0'X`S)BB)9S=8!?0`#O`&33-'?'-+9
XM3``'X`T`0-[>_9GI+-&330`48`8`@*BZ8LRR+=]?)"\4H)R(BB(]'=,V',N^
XMO:KE.:3"O:H'8-G8#1+!+,\(P*C"0:I(O;0(``1E;4VA'=44(=+R@@"P8<OF
XMG9G@'>(J(QRJBM)C3;>KB@"B(1RZ8ISQ[>*HB@"84-;"@<_*C.&F4-8[$MIZ
XM+>'-4-81PN.!O=POSMW"D:6AK=@2OK/"<2`?&ME*CJH),-'"`9Z9O;0)((;"
XM(:VH+=KRG`#2L>,S]+%>3IW",1%&^L_/?,H)(`ME'1%0VLK>N]`'CJH)7JI>
XMC@UEC2*AK=PV;@#H"N2$^IE)?>4&H`">B=W&Z>'#K0!/4-99*J9:S>@*4`<;
XMCMSG+<\*T+@5[NEKO=4!K0!PJN(E'L]DG=X*(`Q%JL`H/>2:?@QES>6`O;0*
XM``]&_M2M">6,O@`R.N4`$-TIK>2GO`#*)>-'#=..V9H1T=NSC.#%[N>F+B\+
XM$)BJCM*$CLT!O0!:A*(KKNBCS=F>4-:MV>'ESMJZ4-:D^N8FOMH4D,S"84V,
XM^NFZ(M04\*#"`=KPO>X5,!'"@2(B\)FLGM,&4`$J8.FD3NL(7P%@KLPBW9I)
XM_O"8(AP[,N;`_O"42NRA;>4/+^K\W.6K70$IB^9BFK$E?YEMOM)PGN_/*B\5
XMT*#"$1&/W<H'(NT-W><X/=7RT@##+NO)C=X!W0#D+>XEONB%W@`@H2NM&>E+
XMVP"]F=^);LWRW`!2HRO69)PM/MP-D`GE+1P07NJ,W@#M+J'-+M=7/P[EC>OY
XM+,\.(*VZ`MNY#O<Y^-(["M#RX@`6JBL1$A$(_>P2K_/4KN!$O_?UL>U#?^T$
XMX`!2@_0HK?3#[0"<;LQC+NFKZ@"-2ZJD^J&97N@.`.ND:DWDBN^@/Z"DBB(5
XM@<Y+"[$$R\P&O[0/4!JDNB,3_?(N?LH/P'ZDFJ43$?B?.:^$CZH]R!4]#^($
XM\`"$7!JEP>TGGOQB>+9)+_N]J;7&COFH^@`6>B!^;?6,_@"P09O=#P",NB.0
XM2;F[2:K"H;!96N,50<O)7S;`&OLJ#?\/D"W9/`".ZLHA;?^<[+O[7_'O']`/
XMP)_<JO^?N?'O']`/D`AXN_\@__X!_0"BO".I_YF:3=;VSZ83D?J?J?)D;?^;
XM`,BIK^;U']`/H,JZLO\G6_\!_0"7NB.ZLO\M6_\!_0"M(!'EO?\S6_\!_0#X
XMNSRZLO^+6_\!_0"Y\,7[7P']']`/`*?3G*RMJ;3U']`/\*(3@2+[__GO']`/
XM\+)HG*RMJ;`R2M[H3U0%'[,$.[;U']`/P+H%X/04T?\!_0"A.<X#X/3MW?\!
XM_0#5T,_&&[7]']`/,,`18;RMJ;?R_`#9P*[3_)GDZ@'/BOXY+/&T#P"&6_\!
XM_0`UZRCGV_\!_0#ED*Q@VYIZ6_\!_0`+*JU@VYHB(/L-2IL'\M%%*OOX(/&:
XM>]%/C:*>B?XY#*AA2ZJO6_\!#0'C["BTV_\!#0'3K,6M*;KU']`0<"#2"K:M
XMB:Q?M/#H3U2D&A'D3:K)6_\!#0$57`#[_[SU']`04!KY_]'HV?\!#0'D3<M]
XM&_P`H%S'@_Y$1:ZJ:ORCT?\!#0'%6P#[S\+U']`00)OYO__56_\!#0$+[[O[
XMO]^Y']`0<#P%'Q%0V\H3,?P&D'6TO[00H.''/>:$/MP0\"Q&'=J2OZH0P+'1
XM2>K8;P`0$`N8J2N8OK00,,"?7?7>N;00\)K"">\U/MP14/"O7?4]+L\1T)O6
XM%"&ES__R'`%?^M<?6O'#'0%F:DW2*J8;/]P1D`N8F:6W#ZA+&P$#;$T3$=J:
XMS>@1\)K69.?VJ?<$(`$H:DV(6OJM'+-Z/NVHJER(??@$(`%B")[NO/ADS=42
XMX`0?[L[D+MF$;0&C6M90+\\2``T$:TT'@OL*>\H2P`W\#.W._IF:._P&P'X"
XM3[`*?</J*?$%K_4AS:T$(`'L$+"`?\PJS:T$(`&6C<<#`/@PS?@3H"L384W?
XM_<RV>L<'<M%;:_`BS=L!^P_JB9D%S_KU7]&_6@`7'>]D#?\3,*^0V:V?Z?"K
XM7]'6$Q$W6_<*W9I3\@^MB:J)[:P7.R7_L)MW9,P@?[%3\@^[>3R:J]_.RA7D
XM.B7_T)H_W]%$-1!$U0%V0%0>0*D`X`&6'1+4"0`A8-\S])H*L8/TF;*$FLSH
XM2:,`P`)B.!JN0%0NX`,:"%=)3D0N4E-$``(:B``^!```.!+1LO4&0`@```P`
XM"`&`5$!`DR).J@@)(F6@@0$!`!`I,F7)P`0,(A()(N1)%2H##U2(:"1*E214
XMJEQQ>"$BDR=!B`QYXL3(P`(8(DX)8J7(S)HW,T0<$@1*2BE%'&H@:1+E0`0;
XM2+X<LF1)D2PW.0Q]TJ1)$"=$'';0680*E"!4D#CT$-&JE"9.(1WX$+&)$:1)
XM(2$(4==(D"D@]8J(>"6)$\!9F.0M,(*P82HH%=\D0;**DR%4K&(E6")BD"I$
XMD@@IPH3)31,1K229(IJT:8(G(E)Y`H7FZ-<%4'B>,D5*6+TI(B+M_1N!"K)4
XM?MJ$9&"%<(IE/V.YV2+BDYY2`$M)<O,%8:1.@AQT"$,G$BE.EA1V<C-&1"1)
XM)JYW*"-B1RJSF\R^.<,Z:RU%S$<0#75=%Z!A#M4PU$M3',@>039$9!@4'PE8
XMP`T1G0587@C@X-AEBC%T4PX?#I>$%C?M\%P34R#AVDT\F.?:##(XU(.,I<E`
```

```
XM@T,^X,A$##4R]X./.(0$A(]/I'53$"X],<1,4&QFP!`D.4&%$$D<$1(155(Q
XM1193A%1$ET]<\>`!1G09!!.!'7!$EY`Y(24275JVA!,K$:1$AE)<F217-RW!
XM)WZTA<3$H.O]9D`3@P)6*',1#<"11VT&4(!]'\WVH`$!<`I`$T\@99@1,=PD
XM```#.!%@>.,1)$``4A%1JEX"0`3`$TN`)]YB`Z!JA&68%4%$8`0,0$!EEZF*
XM!;$#7`K`KY=!@905`Q7K*;1#M%AFM0,<@.P016`15[$*?,MJ7L5F]"RP'1&Q
XM6;$0?(M08`80@&JB_Z%+@+.)$G59M01XFBBN#A'@+0")2E%FF)`00``"'\KT
XM!!,,%U``JJ,^,>L!!1P+P$]4S(J``1C7]$20!Q@`\<<T41$D`@>4;,03,X1T
XMP*LL6UFS7@C(_,2.<B&`,\A`(Y"`SS6$E,#0+2>MEP*H9J><0^4*]^17(2F`
XMLQ0S6?3TP4(($4,.,L!`]<IAYV"#V7(ID(!]0M"``]L'*%`N`&'/L';6"\"M
XM(]T*J!MVV8`W`'<,<V?MP.%E9_T`W'L_'2_>0LS`-@(*1.#WY0I(<+C3F$]P
XM^,Z84W"XR`J,1/D-%RE@`=Q.)Z!`3G@/(03H"@A5NQ`V/*7`4KNS_G14NQ?Y
XM-%L`0"&$$S3E98`"="4_6D^;*@!"ANW^5C=E`$P1-NJ==?^]TW6C)KX0+RN@
XMV[.V,Z_J30H$Q[X06-!$]7'S]\GP\RQ@:L1_LRJ``JJ#MRK\[T1!$J`+_/<?
XM&MUD`:B:RA(R=A,&H.HL4A!/2#)B':M@P2<.88"M,"@>[=RD`:AR2Q,PV(1J
XM-0!G#0*+$9(@F88UP%9(&<(,:UB`!GA,A>$:5P,.E@2O'*$(H/H-`1JPLK\,
XM(0E%?((2&_"VG%GA7"Z\VT2$4(4CAL>%?9-($;AXA%D9H`'JVF(7=W9&PXF1
XMC#4"`!H(5TE.1$%,5"Y&3E0``","``!T"Y1&'2:`"```#``@03`0\(&"@/\"
XMQ@,!J0``$`$30($2T,%$?%`P4H24``0>4'!0P(/($1@R"""`8,(0$0@H4$!0
XM00*`)B(,&"D#,D`)!-(`$!!:`O69$Y("*"#P@=BX`$12IP5O%E6`+V"$FP$5
XM(.`)*"L<1)A0(4,$)R((,`P)E.7X=6O*JFP1(8"!@&Q$"#"@S,0'(2A'?(#@
XM(:B[-@$,(($1V?WK%G'7N&05%_Z*",]6,`$78,4*23,,`%*!!#S0&"B"S%7A
XM+@`$!&C*QPG:HGP8\2LN5+@`X3$+13$^Q1'A*1:."%YM1(!\+DX@O*'QOX``
XM@8L.-P%@Z<D''`>$2W)$1#Y_^ZS-L$!A!`V7@P<`"ABHF8@B)C=0'1$V5.`)
XM?$>$#!4FX'$U5!AQ\&1WG$\RT13<<.\!$%]<@,!!6$1*04(01^`94!AX!2#U
XM'20$H(+*1@DHAE1O#Y:("%(@4M@8(-7!T9-Y6;%FFE<0",C;@Q-4!P9:?CD@
XM(1Z5R9>83\\M\%5TNW$T&)$^-:ED<;K9!`0X/>G7V5=1(K#6`=$UIV5``*#E
XM4V%!P9`C`1Y"`F9R[+DW4T1FTM@9-#)IN0`L8XVW)7@#K+5`<P,4"!N4`UAV
XM&D=\&M@9'A$B\%P"0%Q9:9L+`(I)39T!BJEB(C*UXE(/9@JB987Q]=9=3K4&
XME%GC,:0=1ZZB5!2EJ/CUP:B]998?JA3R5)VMM<*JG:Q"N79K3'YQ`!H(5TE.
XM1$]42"Y&3E0```<"``!T"Z1&,[,`"```#``@03`0\`&"@/\"Q@,!B0``$`$7
XMH$`1T`$2?!>1!%0``@\H."C@140&#0063!O!@$&""A.<C0P'!&P``DA#B)`X
XM0K@9$00J.*AP+@`!`A_1BC`@0`C(`%]`"#!@1$0`H2:@C7@PH4*%3&1.$&!`
XM$76:$PZBJD#(*L"'``\"1"]S(H(T@.W!G/@`P4.`P*L"((#TPMW(%NW5LHB@
XMF(VKP*Q;NCT!/&0*\6'-C6A!+(6T`(]DSY`8:*:*LS$B!%)+XT.$"Q8@M681
XMK4:T$1XBV[<WF@T\."<X2``0S04`CK#>P&KS`@(7N'9@3+(WSJTKG#!#`FH1
XM-.RM`!,H8,!`H<P)"%(!M8BPH9HN'5E+Z(3G8J\M'%YSO,)1`4=W-P%N>*``
XMIQLB8$#`G5'F26<>8]-!`45ITZ&"#6UR)08"%!0J(-R#D.%%6EH;X7`=#A$!
XM`L1H&X%S77%?`1734V0E<.)F#\'!''!H;`0$$'`@@41<#."P(XDY\86#CT0N
XM8%9>C,&`!!#@[,B4DSCT)B./2'0XU$-:+J44<&`$="448$"A44Y@7`<D%%M1
XMR``4:/C(5&Q`]H@&%$!$9",`L$"!0418@"$E9W```@<"C!$%!U%2A9;E`$0R
XMX..#3"&R%11,(1'5F9(B@0-J$>%1%6A#00!'59MQ!-&+7_%X8IZ<V141/I`0
XM!$D"&@A724Y$4U1$+D9.5```5`(``'0+FT:PL0((```,`"!!.!#P08.`_P+&
XM@P&)(`P8`0VP@30@8`,D2/CQPZ@18\2'>)@!P\.`&9Z'`0LP8X8!PT,P8%8&
XM/``'&SPP=+@AXT8GHDLP%G]",@`#0T0P18?"``H)P4H\&D,R"XB`X0"-5@,Z
XM>`B':0)\^`)&@!F1`8.6+\$``Q8Q)$5N[)A1Q/,1#CR&!OBU7<F``PXXX(#I
XM'<KO+`P<9AF9](D!!PX\6+#Q:QF1WUI\#"@^W7M9+@&ZA,VB74IQ[US/H`VX
XM)<`G,>@$9`&0A;0`I=6N`0D@^+D6*08$5#4"`""\*2!@OGD#VLMHMVV(#:5R
XMH\9M+1_3!/AYCHA/[H#ND`B$'>IV`##-BX?"PR;7@,W*Y[^?SPZ?(K[S!3@#
XMXX8ZHF?MD.1''D,(I):94ZFMA(U-Z[$WU5#G)3"8`80PHU%[_C'##CL:<</?
XM@ZK)52!WWMT7H&D%4,->3T.!]YU<`H9(##APX,"!8JGQPU`"_D&20()#(7$2
XM=`:T=R$A/0XP9'D]%G!2>)6=Y=)2:DV(!QA-I=8;4D5A8%9;&&09$00XL&&,
XM(`%54!P!13$0$`0A,<#'2M=!:!^,X_THDF8&4C1G@'7J:2$PZ1E0)DSX8"E@
XM`G\6T-IX!,0'G@$?000'0TXR!@`.8;H'1V[Q,<,@>\1$=*F8/N+##S7H!?05
```

```
XMACZZE6FJ>*XU4Z.MN=E0=_BY&M)():5FE8X!<N!J>X%Z)R1*/GI'BW9(-FL2
XM#%(=.]2C,Q7F%V"#'<"!53!996R`#"U@:I@#B$;1ITV1P0\F;\*`!"'=`AH(
XM5TE.1%1.62Y&3E0``)L!``!T"Z!&6QD`"```#``@02`0\,&!@/\"P@,"B2"0
XM@`M`@0KH`!0XBQ,A*0`"#A@8<`\A+0`$`@C)@`K`@`$""Q(`,"B!F`S8P&3#
XMD!M!W(3($61`!AQ#/@!1$B*X@!)4HB0J$Q"@F*!V:@0"C.&`HU-!E02$50&X
XMDB!`X00!QB)1E.```0N+TR0PB3B_EIRI\2(0B5WO@@)35.1#`"&!`J:[8"L0
XM$$8!8&T`Z##=C5H?QL7(->;%BQD5O'T+"AA4IV)1OFWHN:Y:IUW3`@,$:<!G
XMNRCQ1G5=ER&!KCH)A-8H$1AGE*P+=+5XL?52XJ">UHUZ&ZU%UK0AMP8'<WE'
XMB2C!-`Y;VBO#`K$A%:BN@+E/WM,OQA:K=[UMM',K:UQ-&GCCPX@U\FU(/FC#
XM@O#$=!A%:UU$'G27\93<;@KH=)5%"J;57W*K*9?27FP!U]IO4P'VGT:($<4@
XM:+YE5E5#6#&`448,J(4=)$#!!=%F3D%4%E_YJ5BAC=MU)Q-(0%2G8FM"SA:8
XC;#^QEU%$.`8&DGP;W==83,99%5`"CCGV$Q0L@@.)`2@!&@!(
X`
Xend
SHAR_EOF
chmod 0600 BINARC.UUE || echo "restore of BINARC.UUE fails"
sed 's/^X//' << 'SHAR_EOF' > BLDMFORM.C &&
X#include <stdio.h>
X
Xmain (ac, av)
Xchar    **av;
X{
X    char **ap, *cp;
X    int  xhot, yhot, nplanes;
X    int  maskcolor, datacolor;
X    char *datafn, *maskfn;
X    char *name;
X    short    gemfmt = 0;
X    short    rename = 0;
X
X    if (ac < 2)
X    {
X         printf("usage:  -xypMDmdnG xhot yhot planes mcolor dcolor\n");
X         printf("\t\tmaskfn datafn mformname\n");
X         printf("or:     -dnR datafn newmformname\n");
X         exit(0);
X    }
X
X    xhot = 0;
X    yhot = 0;
X    nplanes = 1;
X    maskcolor = 0;
X    datacolor = 1;
X    maskfn = (char *) 0;
X    datafn = (char *) 0;
X    name = "mform";
X
X    for (ap = av + 1; *ap && **ap == '-'; )
X         for (cp = *ap++ + 1; *cp; ++cp)
X              switch (*cp)
X              {
X              case 'x':
X                   xhot = atoi(*ap++);
```

```
X                    break;
X               case 'y':
X                    yhot = atoi(*ap++);
X                    break;
X               case 'p':
X                    nplanes = atoi(*ap++);
X                    break;
X               case 'M':
X                    maskcolor = atoi(*ap++);
X                    break;
X               case 'D':
X                    datacolor = atoi(*ap++);
X                    break;
X               case 'm':
X                    maskfn = *ap++;
X                    break;
X               case 'd':
X                    datafn = *ap++;
X                    break;
X               case 'n':
X                    name = *ap++;
X                    break;
X               case 'G':
X                    ++gemfmt;
X                    break;
X               case 'R':
X                    ++rename;
X                    break;
X               }
X
X    if (rename)
X         chgname(datafn, name);
X
X    if (gemfmt)
X    {
X         printf("/* GEM Icon Definition: */\n");
X         printf("#define %s_W 0x0010\n", name);
X         printf("#define %s_H 0x0028\n", name);
X         printf("#define %sSIZE 0x0028\n", name);
X         printf("short %s[%sSIZE] =\n{ ", name, name);
X         hexw(5, xhot, yhot, nplanes, maskcolor, datacolor);
X    }
X    else
X    {
X         printf("#ifndef USER_DEF\n#include <gemdefs.h>\n#endif\n");
X         printf("MFORM\t%s[1] = {\n\t%d,\t%d,\t%d,\t%d,\t%d,\n",
X              name, xhot, yhot, nplanes, maskcolor, datacolor);
X    }
X
X    domask(maskfn, gemfmt);
X    if (!gemfmt)
X         fputs("\t },\n", stdout);
X    domask(datafn, gemfmt);
```

```
X
X    if (gemfmt)
X         fputs("0x0000, 0x0000, 0x0000", stdout);
X    else
X         fputs("\t }", stdout);
X    fputs("\n};\n\n", stdout);
X    exit(0);
X}
X
Xdomask (fn, gemfmt)
Xchar    *fn;
X{
X    char line[80];
X    int  h0, h1, h2, h3;
X    FILE *fd;
X
X    if (fd = fopen(fn, "r"))
X    {
X         *line = '\0';
X         while (fgets(line, sizeof(line), fd) && *line != '{');
X         if (*line == '{')
X         {
X              do
X                   if (gemfmt)
X                   {
X                        sscanf(line + 1,
X                             " 0x%x, 0x%x, 0x%x, 0x%x",
X                             &h0, &h1, &h2, &h3);
X                        hexw(4, h0, h1, h2, h3);
X                   }
X                   else
X                        printf("\t %s", line);
X              while (fgets(line, sizeof(line), fd) && *line != '}');
X         }
X         fclose(fd);
X    }
X}
X
Xhexw (count, h0)
X{
X    int  *ap;
X    static   int  col;
X
X    for (ap = &h0; --count >= 0; )
X         printf("0x%04x,%s", *ap++, (++col % 4)? " ": "\n  ");
X}
X
Xchgname (datafn, name)
Xchar    *datafn, *name;
X{
X    FILE *fd;
X    char line[80];
X    register char *cp;
```

```
X
X    if (fd = fopen(datafn, "r"))
X    {
X         while (fgets(line, sizeof(line), fd) != NULL &&
X              strncmp(line, "short", 5))
X              fputs(line, stdout);
X         printf("short %s", name);
X         for (cp = line; *cp && *cp != '['; ++cp);
X         fputs(cp, stdout);
X         while (fgets(line, sizeof(line), fd) != NULL)
X              fputs(line, stdout);
X         fclose(fd);
X    }
X
X    exit(0);
X}
SHAR_EOF
chmod 0600 BLDMFORM.C || echo "restore of BLDMFORM.C fails"
sed 's/^X//' << 'SHAR_EOF' > KERMIT.C &&
X#define EXTRADEBUG
X/*
X *  K e r m i t  File Transfer Utility seriously hacked for local use with uw
X *
X *  Currently allows only one kermit session to exist in one window concurent.
X *
X *  Adapted from UNIX Kermit, Columbia University, 1981, 1982, 1983
X *      Bill Catchings, Bob Cattani, Chris Maio, Frank da Cruz, Alan Crosswell
X *
X *  Also:   Jim Guyton, Rand Corporation
X *          Walter Underwood, Ford Aerospace
X *
X */
X
X/*
X * March 28, 1988
X *      Hacked into form usable in uw.  Converted fsm's to be driven
X *      by received packet events.  Rewrote rpack routine.
X */
X
X#include <obdefs.h>
X#include <gemdefs.h>
X#include <osbind.h>
X#include <stdio.h>          /* Standard UNIX definitions */
X#include <time.h>
X#include "wind.h"
X#include "windefs.h"
X
X#define error printmsg
X#define chari int     /* items of type chari should be char, but, mwc
X                   insists on adjusting them to int anyway */
X/* Symbol Definitions */
X
X#define MAXPACKSIZ  94      /* Maximum packet size */
```

```
X#define SOH         1       /* Start of header */
X#define CR          13      /* ASCII Carriage Return */
X#define SP          32      /* ASCII space */
X#define DEL         127     /* Delete (rubout) */
X
X#define MAXTRY      10      /* Times to retry a packet */
X#define MYQUOTE     '#'     /* Quote character I will use */
X#define MYPAD       0       /* Number of padding characters I will need */
X#define MYPCHAR     0       /* Padding character I need (NULL) */
X
X#define MYEOL       '\r'    /* End-Of-Line character I need */
X
X#define MYTIME      10      /* Seconds after which I should be timed out */
X#define MAXTIM      60      /* Maximum timeout interval */
X#define MINTIM      10      /* Minumum timeout interval */
X
X#ifndef TRUE
X#define TRUE        -1      /* Boolean constants */
X#endif
X#define FALSE       0
X#define GETPACK     2     /* return constant */
X
X/* Macro Definitions */
X
X/*
X *  f l u s h i n p u t
X *
X *  Dump all pending input to clear stacked up NACK's.
X */
X
X#define flushinput()  /* no explicit call to flush needed for uw */
X
X/*
X * tochar: converts a control character to a printable one by adding a space.
X *
X * unchar: undoes tochar.
X *
X * ctl:    converts between control characters and printable characters by
X *         toggling the control bit (ie. ^A becomes A and A becomes ^A).
X */
X#define tochar(ch)  ((ch) + ' ')
X#define unchar(ch)  ((ch) - ' ')
X#define ctl(ch)     ((ch) ^ 64 )
X#define abs(exp) (((exp) >= 0) ? (exp) : -(exp))
X
X
X/* Global Variables */
X
Xextern int   mouse;                 /* is mouse visible ? */
Xextern  struct   wi_str   w[];
X
Xint     size,               /* Size of present data */
X        rpsiz,              /* Maximum receive packet size */
```

```
X        spsiz,              /* Maximum send packet size */
X        pad,                /* How much padding to send */
X        reqtimint,          /* timeout interval I request */
X        timint,             /* Timeout for foreign host on sends */
X        n,                  /* Packet number */
X        numtry,             /* Times this packet retried */
X        oldtry,             /* Times previous packet retried */
X        image,              /* -1 means 8-bit mode */
X        debug,              /* indicates level of debugging output (0=none) */
X        filnamcnv,          /* -1 means do file name case conversions */
X        filecount,          /* Number of files left to send */
X    kermwdes,    /* Window descripter for kermit window */
X    kermport;    /* port number for kermit window */
X
Xclock_t timestamp;        /* Time last packet was received */
X
Xchar    sflg, rflg;         /* flags for RECEIVE, SEND */
Xchar    state,              /* Present state of the automaton */
X        padchar,            /* Padding character to send */
X        eol,                /* End-Of-Line character to send */
X        quote,              /* Quote character in incoming data */
X        **filelist,         /* List of files to be sent */
X    *filnam,            /* Current file name */
X        recpkt[MAXPACKSIZ+1], /* Receive packet buffer */
X        packet[MAXPACKSIZ+1]; /* Packet buffer */
X
XFILE    *fp,                /* File pointer for current disk file */
X        *log;               /* File pointer for Logfile */
X
X
X/*
X *  kerminit
X *
X *  initalization routine - initalize and dispatch to the appropriate routine.
X */
X
Xint kerminit(curwin)
Xint curwin;
X{
X    OBJECT *obj_tmp;
X    static char path[40] = ".\*.*";
X    static char file[40] = "";
X    static char filename[80];       /* space for file and directory */
X    int butt;
X
X/*  Initialize these values and hope the first packet will get across OK */
X
X    if (kermwdes && w[kermwdes].kerm_act) return (-1);
X    kermwdes = curwin;
X                            /* If kermit already active, return. */
X    w[kermwdes].kerm_act = TRUE;     /* mark current window to send output
X                               to kermit instead of emulator */
X    kermport = find_port(curwin);
```

```
X    eol = CR;                          /* EOL for outgoing packets */
X    quote = '#';                       /* Standard control-quote char "#" */
X    pad = 0;                           /* No padding */
X    padchar = '\0';                    /* Use null if any padding wanted */
X    timint = MYTIME;                   /* default timint */
X
X
X/*
X * Set direction and paramaters debug, filnamcnv, image with dialog
X */
X    rsrc_gaddr(R_TREE, KERMPARM, &obj_tmp);
X    if (obj_tmp[ASCIIMOD].ob_state == NORMAL &&
X      obj_tmp[IMAGEMOD].ob_state == NORMAL)
X    {                      /* initalize object and statics */
X        objc_change(obj_tmp, ASCIIMOD, 0, 0, 0, 0, 0, SELECTED, 0);
X        objc_change(obj_tmp, CONVNAME, 0, 0, 0, 0, 0, SELECTED, 0);
X        objc_change(obj_tmp, RECFILE, 0, 0, 0, 0, 0, SELECTED, 0);
X
X    path[0] = Dgetdrv() + 'a';
X    path[1] = ':';
X    Dgetpath(path+2, 0);
X    strcat(path, "\\*.*");
X    }
X    if ((butt = s_dial(KERMPARM, 3)) == KERMEXIT) return(kermterm());
X    if (obj_tmp[DEBUGENA].ob_state == SELECTED)
X    {
X        debug = obj_tmp[DEBUG1].ob_state + obj_tmp[DEBUG2].ob_state
X    + obj_tmp[DEBUG3].ob_state;
X    }
X    if (butt == RECFILE) obj_tmp[RECFILE].ob_state = SELECTED;
X    if (butt == SENDFILE) obj_tmp[SENDFILE].ob_state = SELECTED;
X    rflg = obj_tmp[RECFILE].ob_state;
X    sflg = obj_tmp[SENDFILE].ob_state;
X    image = (obj_tmp[IMAGEMOD].ob_state == SELECTED);
X    filnamcnv = obj_tmp[CONVNAME].ob_state; /* conversion for UNIX systems */
X
X/* All set up, now execute the command that was given. */
X
X    if (debug)
X    {
X        printmsg("debuging level = %d\n",debug);
X
X        if (sflg) printmsg("Send command\n");
X        if (rflg) printmsg("Receive command\n");
X    }
X
X    if (sflg)                          /* Send command */
X    {
X        extern char * rindex();
X
X    if (!mouse)
X    {
X        graf_mouse(M_ON, NULL);
```

```
X        ++mouse;
X    }
X        fsel_input(path, file, &butt);
X    if (! butt) return(kermterm());
X    strcpy(filename, path);
X    if (debug > 2) printmsg("directory %s", filename);
X    filnam = rindex(filename, '\\');
X    if (filnam) *(++filnam) = '\0';
X    strcat(filename, file);
X        filnam = filename;      /* Get file to send */
X        if (debug > 2) printmsg("sending %s",filnam);
X        fp = NULL;                      /* Indicate no file open yet */
X        filelist = NULL;                /* Set up the rest of the file list */
X        filecount = 0;                  /* Number of files left to send */
X    state = 'S';                    /* Send initiate is the start state */
X    n = 0;                          /* Initialize message number */
X    numtry = 0;                     /* Say no tries yet */
X        w[kermwdes].kerm_act = TRUE;/* mark current window to send output
X                                to kermit instead of emulator */
X    timestamp = clock();        /* reset timer */
X    sendsw();
X    }
X
X    if (rflg)                           /* Receive command */
X    {
X    state = 'R';                    /* Receive-Init is the start state */
X    n = 0;                          /* Initialize message number */
X    numtry = 0;                     /* Say no tries yet */
X    timestamp = clock();        /* reset timer */
X    }
X
X    return(0);
X}
X
X/*
X * kermterm - Terminate kermit and tell uw not to call us again.
X */
Xkermterm()
X{
X    sflg = rflg = 0;
X    w[kermwdes].kerm_act = FALSE;
X    kermwdes = 0;
X}
X
Xkermtimchk()
X{
X    if (!w[kermwdes].kerm_act)  /* is kermit active? */
X    {
X    kermwdes = 0;
X    return;
X    }
X    if ((clock() - timestamp) / CLK_TCK > timint)  /* timeout ? */
X    {
```

```
X    rpack(NULL, "", "");    /* tell FSM about time out */
X    }
X}
X
X/*
X *  s e n d s w
X *
X *  Sendsw is the state table switcher for sending files.  It loops until
X *  either it finishes, or an error is encountered.  The routines called
X *  by sendsw are responsible for changing the state.
X *
X */
X
Xint sendsw()
X{
X    chari sinit(), sfile(), sdata(), seof(), sbreak(), nstate;
X
X    nstate = state;
X    while(TRUE)                         /* Do this as long as necessary */
X    {
X        if (debug) printmsg("sendsw state: %c  nstate: %c\n",state,nstate);
X    while (nstate != 'I')
X    {
X            switch(nstate)
X            {
X                case 'S':   nstate = sinit();  break; /* Send-Init */
X                case 'F':   nstate = sfile();  break; /* Send-File */
X                case 'D':   nstate = sdata();  break; /* Send-Data */
X                case 'Z':   nstate = seof();   break; /* Send-End-of-File */
X                case 'B':   nstate = sbreak(); break; /* Send-Break */
X            }
X        switch(nstate)
X        {
X                case 'C':   printmsg("\aDone.\a");
X                 return (TRUE);           /* Complete */
X                case 'A':   printmsg("Send Failed.");
X                 return (FALSE);          /* "Abort" */
X        }
X        if (nstate != 'I')
X        {
X            state = nstate;
X            if (numtry++ > MAXTRY) return(FALSE); /* If too many tries, give up
*/
X        }
X    }
X    return(GETPACK);
X    }
X}
X
X
X/*
X *  s i n i t
X *
```

```
X *  Send Initiate: send this host's parameters and get other side's back.
X */
X
Xchari sinit()
X{
X    int num, len;                      /* Packet number, length */
X    static int sent = 0;
X
X    if (! sent) {
X    reqtimint = MYTIME;
X    spar(packet);                  /* Fill up init info packet */
X    flushinput();                  /* Flush pending input */
X    spack('S',n,6,packet);         /* Send an S packet */
X
X    sent = 1;
X    return('I');           /* get packet to read */
X    }
X    sent = 0;
X    switch(rpack(&len,&num,recpkt))     /* What was the reply? */
X    {
X        case 'N':  return(state);       /* NAK, try it again */
X
X        case 'Y':                       /* ACK */
X            if (n != num)               /* If wrong ACK, stay in S state */
X                return(state);          /* and try again */
X            rpar(recpkt);               /* Get other side's init info */
X
X            if (eol == 0) eol = '\n';   /* Check and set defaults */
X            if (quote == 0) quote = '#';
X            if(abs(timint - reqtimint) < 3) timint = reqtimint + 3;
X            /* guarentee diference of > 3 for sounds for timint */
X
X            numtry = 0;                 /* Reset try counter */
X            n = (n+1)%64;               /* Bump packet count */
X            return('F');                /* OK, switch state to F */
X
X        case 'E':                       /* Error packet received */
X            prerrpkt(recpkt);           /* Print it out and */
X            return('A');                /* abort */
X
X        case FALSE: return(state);      /* Receive failure, try again */
X
X        default: return('A');           /* Anything else, just "abort" */
X   }
X }
X
X
X/*
X *  s f i l e
X *
X *  Send File Header.
X */
X
```

```
Xchari sfile()
X{
X    int num, len;                      /* Packet number, length */
X    char filnam1[50],                  /* Converted file name */
X        *newfilnam,                    /* Pointer to file name to send */
X        *cp;                           /* char pointer */
X    static int sent = 0;
X
X    if (! sent) {
X        if (fp == NULL)                    /* If not already open, */
X        {   if (debug) printmsg("   Opening %s for sending.\n",filnam);
X            fp = fopen(filnam, image ? "rb" : "r");         /* open the file to
be sent */
X            if (fp == NULL)                /* If bad file pointer, give up */
X            {
X                error("Cannot open file %s",filnam);
X                return('A');
X            }
X        }
X
X        strcpy(filnam1, filnam);           /* Copy file name */
X        newfilnam = cp = filnam1;
X        while (*cp != '\0')                /* Strip off all leading directory */
X            if (*cp++ == '\\')              /* names (ie. up to the last /). */
X                newfilnam = cp;
X
X        if (filnamcnv)                     /* Convert lower case to upper  */
X            for (cp = newfilnam; *cp != '\0'; cp++)
X                if (*cp >= 'a' && *cp <= 'z')
X                    *cp ^= 040;
X
X        len = cp - newfilnam;              /* Compute length of new filename */
X
X        printmsg("Sending %s as %s",filnam,newfilnam);
X
X        spack('F',n,len,newfilnam);        /* Send an F packet */
X    sent = 1;
X
X    return('I');           /* get packet to read */
X    }
X    sent = 0;
X    switch(rpack(&len,&num,recpkt))     /* What was the reply? */
X    {
X        case 'N':                      /* NAK, just stay in this state, */
X            num = (--num<0 ? 63:num);  /* unless it's NAK for next packet */
X            if (n != num)              /* which is just like an ACK for */
X                return(state);         /* this packet so fall thru to... */
X
X        case 'Y':                      /* ACK */
X            if (n != num) return(state); /* If wrong ACK, stay in F state */
X            numtry = 0;                /* Reset try counter */
X            n = (n+1)%64;              /* Bump packet count */
X            size = bufill(packet);     /* Get first data from file */
```

```
X            return('D');               /* Switch state to D */
X
X        case 'E':                      /* Error packet received */
X            prerrpkt(recpkt);          /* Print it out and */
X            return('A');               /* abort */
X
X        case FALSE: return(state);     /* Receive failure, stay in F state */
X
X        default:    return('A');       /* Something else, just "abort" */
X    }
X}
X
X
X/*
X *  s d a t a
X *
X *  Send File Data
X */
X
Xchari sdata()
X{
X    int num, len;                      /* Packet number, length */
X    static int sent = 0;
X
X    if (! sent) {
X        spack('D',n,size,packet);           /* Send a D packet */
X    sent = 1;
X    return('I');           /* get packet to read */
X    }
X    sent = 0;
X    switch(rpack(&len,&num,recpkt))    /* What was the reply? */
X    {
X        case 'N':                      /* NAK, just stay in this state, */
X            num = (--num<0 ? 63:num);  /* unless it's NAK for next packet */
X            if (n != num)              /* which is just like an ACK for */
X                return(state);         /* this packet so fall thru to... */
X
X        case 'Y':                      /* ACK */
X            if (n != num) return(state); /* If wrong ACK, fail */
X            numtry = 0;                /* Reset try counter */
X            n = (n+1)%64;              /* Bump packet count */
X            if ((size = bufill(packet)) == EOF) /* Get data from file */
X                return('Z');           /* If EOF set state to that */
X            return('D');               /* Got data, stay in state D */
X
X        case 'E':                      /* Error packet received */
X            prerrpkt(recpkt);          /* Print it out and */
X            return('A');               /* abort */
X
X        case FALSE: return(state);     /* Receive failure, stay in D */
X
X        default:    return('A');       /* Anything else, "abort" */
X    }
```

```
X}
X
X
X/*
X *  s e o f
X *
X *  Send End-Of-File.
X */
X
Xchari seof()
X{
X    int num, len;                      /* Packet number, length */
X    static int sent = 0;
X
X    if (! sent) {
X        spack('Z',n,0,packet);             /* Send a 'Z' packet */
X
X    sent = 1;
X    return('I');           /* get packet to read */
X    }
X    sent = 0;
X    switch(rpack(&len,&num,recpkt))     /* What was the reply? */
X    {
X        case 'N':                      /* NAK, just stay in this state, */
X            num = (--num<0 ? 63:num);  /* unless it's NAK for next packet, */
X            if (n != num)              /* which is just like an ACK for */
X                return(state);         /* this packet so fall thru to... */
X
X        case 'Y':                      /* ACK */
X            if (n != num) return(state); /* If wrong ACK, hold out */
X            numtry = 0;                /* Reset try counter */
X            n = (n+1)%64;              /* and bump packet count */
X            if (debug) printmsg("   Closing input file %s, ",filnam);
X            fclose(fp);                /* Close the input file */
X            fp = NULL;                 /* Set flag indicating no file open */
X
X            if (debug) printmsg("looking for next file...\n");
X            if (gnxtfl() == FALSE)     /* No more files go? */
X                return('B');           /* if not, break, EOT, all done */
X            if (debug) printmsg("   New file is %s\n",filnam);
X            return('F');               /* More files, switch state to F */
X
X        case 'E':                      /* Error packet received */
X            prerrpkt(recpkt);          /* Print it out and */
X            return('A');               /* abort */
X
X        case FALSE: return(state);     /* Receive failure, stay in Z */
X
X        default:    return('A');       /* Something else, "abort" */
X    }
X}
X
X
```

```
X/*
X *  s b r e a k
X *
X *  Send Break (EOT)
X */
X
Xchari sbreak()
X{
X    int num, len;                      /* Packet number, length */
X    static int sent = 0;
X
X    if (! sent) {
X        spack('B',n,0,packet);             /* Send a B packet */
X
X    sent = 1;
X    return('I');           /* get packet to read */
X    }
X    sent = 0;
X    switch (rpack(&len,&num,recpkt))    /* What was the reply? */
X    {
X        case 'N':                      /* NAK, just stay in this state, */
X            num = (--num<0 ? 63:num);  /* unless NAK for previous packet, */
X            if (n != num)              /* which is just like an ACK for */
X                return(state);         /* this packet so fall thru to... */
X
X        case 'Y':                      /* ACK */
X            if (n != num) return(state); /* If wrong ACK, fail */
X            numtry = 0;                /* Reset try counter */
X            n = (n+1)%64;              /* and bump packet count */
X            return('C');               /* Switch state to Complete */
X
X        case 'E':                      /* Error packet received */
X            prerrpkt(recpkt);          /* Print it out and */
X            return('A');               /* abort */
X
X        case FALSE: return(state);     /* Receive failure, stay in B */
X
X        default:    return ('A');      /* Other, "abort" */
X   }
X}
X
X
X/*
X *  r e c s w
X *
X *  This is the state table switcher for receiving files.
X */
X
X
X
X
Xrecsw()
X{
```

```
X    chari rinit(), rfile(), rdata();     /* Use these procedures */
X
X    if (debug) printmsg(" recsw state: %c\n",state);
X    switch(state)                   /* Do until done */
X    {
X        case 'R':   state = rinit(); break; /* Receive-Init */
X        case 'F':   state = rfile(); break; /* Receive-File */
X        case 'D':   state = rdata(); break; /* Receive-Data */
X    }
X    if (numtry++ > MAXTRY) return(FALSE); /* If too many tries, give up */
X    switch(state)                   /* Do until done */
X    {
X        case 'C':   printmsg("\aDone.\a");
X            return(TRUE);           /* Complete state */
X        case 'A':   printmsg("Receive Failed.");
X            return(FALSE);          /* "Abort" state */
X    }
X    return(GETPACK);               /* get the next packet */
X}
X
X
X/*
X *  r i n i t
X *
X *  Receive Initialization
X */
X
Xchari rinit()
X{
X    int len, num;                      /* Packet length, number */
X
X    switch(rpack(&len,&num,packet))    /* Get a packet */
X    {
X        case 'S':                      /* Send-Init */
X            rpar(packet);              /* Get the other side's init data */
X            if (timint > MYTIME + 3) reqtimint = timint - 4;
X            else reqtimint = timint + 4;
X            spar(packet);              /* Fill up packet with my init info */
X            flushinput();              /* get rid of unwanted nak's */
X            spack('Y',n,6,packet);     /* ACK with my parameters */
X            oldtry = numtry;           /* Save old try count */
X            numtry = 0;                /* Start a new counter */
X            n = (n+1)%64;              /* Bump packet number, mod 64 */
X            return('F');               /* Enter File-Receive state */
X
X        case 'E':                      /* Error packet received */
X            prerrpkt(recpkt);          /* Print it out and */
X            return('A');               /* abort */
X
X        case FALSE:                    /* Didn't get packet */
X            spack('N',n,0,NULL);       /* Return a NAK */
X            return(state);             /* Keep trying */
X
```

```
X        default:     return('A');       /* Some other packet type, "abort" */
X    }
X}
X
X
X/*
X *  r f i l e
X *
X *  Receive File Header
X */
X
Xchari rfile()
X{
X    int num, len;                       /* Packet number, length */
X    char filnam1[50];                   /* Holds the converted file name */
X
X    switch(rpack(&len,&num,packet))     /* Get a packet */
X    {
X        case 'S':                       /* Send-Init, maybe our ACK lost */
X            if (oldtry++ > MAXTRY) return('A'); /* If too many tries "abort" */
X            if (num == ((n==0) ? 63:n-1)) /* Previous packet, mod 64? */
X            {                           /* Yes, ACK it again with  */
X                spar(packet);           /* our Send-Init parameters */
X                spack('Y',num,6,packet);
X                numtry = 0;             /* Reset try counter */
X                return(state);          /* Stay in this state */
X            }
X            else return('A');           /* Not previous packet, "abort" */
X
X        case 'Z':                       /* End-Of-File */
X            if (oldtry++ > MAXTRY) return('A');
X            if (num == ((n==0) ? 63:n-1)) /* Previous packet, mod 64? */
X            {                           /* Yes, ACK it again. */
X                spack('Y',num,0,NULL);
X                numtry = 0;
X                return(state);          /* Stay in this state */
X            }
X            else return('A');           /* Not previous packet, "abort" */
X
X        case 'F':                       /* File Header (just what we want) */
X            if (num != n) return('A');  /* The packet number must be right */
X            strcpy(filnam1, packet);    /* Copy the file name */
X
X            if (filnamcnv)              /* Convert upper case to lower */
X                for (filnam=filnam1; *filnam != '\0'; filnam++)
X                    if (*filnam >= 'A' && *filnam <= 'Z')
X                        *filnam |= 040;
X
X            if ((fp=fopen(filnam1, image ? "wb" : "w"))==NULL) /* Try to open a
new file */
X            {
X                error("Cannot create %s",filnam1); /* Give up if can't */
X                return('A');
```

```
X            }
X            else                        /* OK, give message */
X
X                printmsg("Receiving %s as %s",packet,filnam1);
X
X            spack('Y',n,0,NULL);        /* Acknowledge the file header */
X            oldtry = numtry;            /* Reset try counters */
X            numtry = 0;                 /* ... */
X            n = (n+1)%64;               /* Bump packet number, mod 64 */
X            return('D');                /* Switch to Data state */
X
X        case 'B':                       /* Break transmission (EOT) */
X            if (num != n) return ('A'); /* Need right packet number here */
X            spack('Y',n,0,NULL);        /* Say OK */
X            return('C');                /* Go to complete state */
X
X        case 'E':                       /* Error packet received */
X            prerrpkt(recpkt);           /* Print it out and */
X            return('A');                /* abort */
X
X        case FALSE:                     /* Didn't get packet */
X            spack('N',n,0,NULL);        /* Return a NAK */
X            return(state);              /* Keep trying */
X
X        default:    return ('A');       /* Some other packet, "abort" */
X    }
X}
X
X
X/*
X *  r d a t a
X *
X *  Receive Data
X */
X
Xchari rdata()
X{
X    int num, len;                       /* Packet number, length */
X
X    switch(rpack(&len,&num,packet))     /* Get packet */
X    {
X        case 'D':                       /* Got Data packet */
X#ifdef EXTRADEBUG
X        if (debug > 2) printmsg("Got Data Packet");
X#endif
X            if (num != n)               /* Right packet? */
X            {                           /* No */
X                if (oldtry++ > MAXTRY)
X                    return('A');        /* If too many tries, abort */
X                if (num == ((n==0) ? 63:n-1)) /* Else check packet number */
X                {                       /* Previous packet again? */
X                    spack('Y',num,6,packet); /* Yes, re-ACK it */
X                    numtry = 0;         /* Reset try counter */
```

```
X                    return(state);      /* Don't write out data! */
X                }
X                else return('A');       /* sorry, wrong number */
X            }
X            /* Got data with right packet number */
X            spack('Y',n,0,NULL);        /* Acknowledge the packet */
X#ifdef EXTRADEBUG
X        if (debug > 2) printmsg("Calling bufemp");
X#endif
X            bufemp(packet,len);         /* Write the data to the file */
X            oldtry = numtry;            /* Reset the try counters */
X            numtry = 0;                 /* ... */
X            n = (n+1)%64;               /* Bump packet number, mod 64 */
X            return('D');                /* Remain in data state */
X
X        case 'F':                       /* Got a File Header */
X            if (oldtry++ > MAXTRY)
X                return('A');            /* If too many tries, "abort" */
X            if (num == ((n==0) ? 63:n-1)) /* Else check packet number */
X            {                           /* It was the previous one */
X                spack('Y',num,0,NULL);  /* ACK it again */
SHAR_EOF
echo "End of part 1, continue with part 2"
echo "2" > s2_seq_.tmp
exit 0


From -7879728040587831079
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,119c0072861f10,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-02 14:06:35 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!csd4.milw.wisc.edu!bionet!agate!
saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i020: uw -- Unix Windows windowing terminal software  part02/05
Keywords: shar
Message-ID: <6552@saturn.ucsc.edu>
Date: 2 Mar 89 22:06:35 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1371
Approved: koreth@ssyx.ucsc.edu

Submitted-by: sun.com!laidbak!katzung (Brian Katzung)
Posting-number: Volume 2, Issue 20
Archive-name: uw/part02

#!/bin/sh
# this is part 2 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file KERMIT.C continued
#
```

```
CurArch=2
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> KERMIT.C
X                numtry = 0;             /* Reset try counter */
X                return(state);          /* Stay in Data state */
X            }
X            else return('A');           /* Not previous packet, "abort" */
X
X        case 'Z':                       /* End-Of-File */
X            if (num != n) return('A');  /* Must have right packet number */
X            spack('Y',n,0,NULL);        /* OK, ACK it. */
X            fclose(fp);                 /* Close the file */
X            n = (n+1)%64;               /* Bump packet number */
X            return('F');                /* Go back to Receive File state */
X
X        case 'E':                       /* Error packet received */
X            prerrpkt(recpkt);           /* Print it out and */
X            return('A');                /* abort */
X
X        case FALSE:                     /* Didn't get packet */
X            spack('N',n,0,NULL);        /* Return a NAK */
X            return(state);              /* Keep trying */
X
X        default:     return('A');       /* Some other packet, "abort" */
X    }
X}
X
X
X/*
X *      KERMIT utilities.
X */
X
X
X/*
X *  s p a c k
X *
X *  Send a Packet
X */
X
Xspack(type,num,len,data)
Xchari type;
Xchar  *data;
Xint num, len;
X{
X    int i;                             /* Character loop counter */
```

```
X    char chksum, buffer[100];          /* Checksum, packet buffer */
X    register char *bufp;               /* Buffer pointer */
X
X    if (debug>1)                       /* Display outgoing packet */
X    {
X        printmsg("  spack type: %c\n",type);
X        printmsg("         num:  %d\n",num);
X        printmsg("         len:  %d\n",len);
X        if (data != NULL)
X    {
X            data[len] = '\0';          /* Null-terminate data to print it */
X            printmsg("        data: \"%s\"\n",data);
X    }
X    }
X
X    bufp = buffer;                     /* Set up buffer pointer */
X    for (i=1; i<=pad; i++) proto_out(kermport,&padchar,1); /* Issue any padding
*/
X
X    *bufp++ = SOH;                     /* Packet marker, ASCII 1 (SOH) */
X    *bufp++ = tochar(len+3);           /* Send the character count */
X    chksum  = tochar(len+3);           /* Initialize the checksum */
X    *bufp++ = tochar(num);             /* Packet number */
X    chksum += tochar(num);             /* Update checksum */
X    *bufp++ = type;                    /* Packet type */
X    chksum += type;                    /* Update checksum */
X
X    for (i=0; i<len; i++)              /* Loop for all data characters */
X    {
X        *bufp++ = data[i];             /* Get a character */
X        chksum += data[i];             /* Update checksum */
X    }
X    chksum = (((chksum&0300) >> 6)+chksum)&077; /* Compute final checksum */
X    *bufp++ = tochar(chksum);          /* Put it in the packet */
X    *bufp = eol;                       /* Extra-packet line terminator */
X    proto_out(kermport, buffer, bufp-buffer+1);
X                           /* Send the packet */
X#ifdef EXTRADEBUG
X    if (debug > 2) printmsg("Returning from spack");
X#endif
X}
X
X/*
X *  r p a c k
X *
X *  Read a Packet
X *  This routine is called both to buffer data from the host to the kermit
X *  window and to retreive those buffers once they contain a packet.  This
X *  routine calls the kermit state machine when it has a complete packet.
X *  The kermit state machine in turn calls this routine to retreive the packet.
X *  Thus, this routine is indirectly recursive to two levels.  Receive timeouts
X *  and bad packets cause FALSE to be returned to the state machine.  The uw
X *  control loop calls us with num == NULL to pass us a string of data from
```

```
X *  the host.  The uw control loop calls us with len == NULL if the timout
X *  counter (kerm_time) exceeds timint without us reseting it.
X */
X#ifdef EXTRADEBUG
X#define getnext {t = *dptr++; \
X            if (debug>2) printmsg( "char %d state %c", t, recstate);\
X        if ((t & 0177) == SOH) { \
X             recstate = 'L'; /* restart packet collection */\
X             break; } \
X        if (!image) t &= 0177; /* Handle parity */ \
X        if (t == 0) break; /* end of this input buffer */ }
X#else
X#define getnext {t = *dptr++; \
X        if ((t & 0177) == SOH) { \
X             recstate = 'L'; /* restart packet collection */\
X             break; } \
X        if (!image) t &= 0177; /* Handle parity */ \
X        if (t == 0) break; /* end of this input buffer */ }
X#endif
Xrpack(len,num,data)
Xint *len, *num;                         /* Packet length, number */
Xchar *data;                             /* Packet data */
X{
X    static int i;                      /* Data character number */
X    static int leng;           /* length of data packet in buffer */
X    static char type;          /* packet type */
X    static int numb;           /* number of data packet in buffer */
X    static int timedout = 0;        /* did we time out? */
X    char t;                            /* Current input character */
X    static char cchksum,                       /* Our (computed) checksum */
X        rchksum;                       /* Checksum received from other host */
X    char *dptr;                /* pointer to next data byte */
X    static char mybuf[MAXPACKSIZ];  /* buffer used to acumulate packet */
X    static char recstate = 'S';     /* start with S state */
X
X    if (len == NULL) {/* timed out */
X        timedout = TRUE;
X    if (sflg)
X    {
X        if (sendsw() != GETPACK)    /* Send the file(s) */
X                return(kermterm());
X    }
X    else if (rflg)
X    {
X            if (recsw() != GETPACK)     /* Receive the file(s) */
X                return(kermterm());
X    }
X        return (0);
X    }
X    if (len != NULL && num != NULL) /* received packet or timed out */
X    {
X    *len = leng;
X    *num = numb;
```

```
X    timestamp = clock();        /* reset timer */
X    if (timedout)
X    {
X        printmsg("Timeout.");
X        timedout = 0;
X        return(FALSE);
X    }
X    for (i=0; i<leng; i++)
X        data[i] = mybuf[i];
X        if (data != NULL)
X            data[*len] = '\0';          /* Null-terminate data to print it */
X        if (debug>1)                    /* Display incoming packet */
X        {
X            printmsg("  rpack type: %c\n",type);
X            printmsg("         num:  %d\n",*num);
X            printmsg("         len:  %d\n",*len);
X            if (data != NULL)
X                printmsg("        data: \"%s\"\n",data);
X        }
X                                        /* Fold in bits 7,8 to compute */
X        cchksum = (((cchksum&0300) >> 6)+cchksum)&077; /* final checksum */
X
X        if (cchksum != rchksum) return(FALSE);
X
X        return(type);                       /* All OK, return packet type */
X    }
X    /* num == NULL so collect data for kermit */
X    dptr = data;
X    t = -1;
X    while (t)
X    {
X      switch(recstate)
X      {
X        case 'S':
X        while (1)         /* Wait for packet header */
X            getnext;
X        break;
X    case 'L':          /* get packet length */
X            getnext;            /* get next character */
X            cchksum = t;                    /* Start the checksum */
X            leng = unchar(t)-3;             /* Character count */
X        if (leng > MAXPACKSIZ)
X            leng = MAXPACKSIZ;
X        i = 0;
X        recstate = 'N';
X        /* Fall Through */
X
X    case 'N':              /* get packet number */
X            getnext
X            cchksum = cchksum + t;          /* Update checksum */
X            numb = unchar(t);               /* Packet number */
X        recstate = 'T';
X        /* Fall Through */
```

```
X
X    case 'T':              /* get packet type */
X             getnext;
X             cchksum = cchksum + t;          /* Update checksum */
X             type = t;                       /* Packet type */
X        recstate = 'D';
X        /* Fall Through */
X
X    case 'D':              /* get packet data */
X             for (; i<leng; i++)             /* The data itself, if any */
X             {                               /* Loop for character count */
X                 getnext;
X                 cchksum = cchksum + t;      /* Update checksum */
X                 mybuf[i] = t;                /* Put it in the data buffer */
X             }
X        if (i == leng)
X            recstate = 'C';
X        break;
X
X    case 'C':
X             mybuf[leng] = 0;                 /* Mark the end of the data */
X             getnext;
X             rchksum = unchar(t);            /* Convert to numeric */
X        recstate = 'E';             /* Done with packet receipt */
X        t = 0;                  /* Dispose of remaining data */
X      }
X    }
X    if (recstate == 'E') {
X        recstate = 'S';
X    if (sflg)
X    {
X        if (sendsw() != GETPACK)        /* Send the file(s) */
X                return(kermterm());
X    }
X    else if (rflg)
X    {
X             if (recsw() != GETPACK)         /* Receive the file(s) */
X                 return(kermterm());
X    }
X    }
X    return(0);
X}
X
X
X/*
X *  b u f i l l
X *
X *  Get a bufferful of data from the file that's being sent.
X *  Only control-quoting is done; 8-bit & repeat count prefixes are
X *  not handled.
X */
X
Xbufill(buffer)
```

```
Xchar buffer[];                         /* Buffer */
X{
X    int i,                             /* Loop index */
X        t;                             /* Char read from file */
X    char t7;                           /* 7-bit version of above */
X
X    i = 0;                             /* Init data buffer pointer */
X    while((t = getc(fp)) != EOF)       /* Get the next character */
X    {
X        t7 = t & 0177;                 /* Get low order 7 bits */
X
X        if (t7 < SP || t7==DEL || t7==quote) /* Does this char require */
X        {                                    /* special handling? */
X            if (t=='\n' && !image)
X            {                          /* Do LF->CRLF mapping if !image */
X                buffer[i++] = quote;
X                buffer[i++] = ctl('\r');
X            }
X            buffer[i++] = quote;       /* Quote the character */
X            if (t7 != quote)
X            {
X                t = ctl(t);            /* and uncontrolify */
X                t7 = ctl(t7);
X            }
X        }
X        if (image)
X            buffer[i++] = t;           /* Deposit the character itself */
X        else
X            buffer[i++] = t7;
X
X        if (i >= spsiz-8) return(i);   /* Check length */
X    }
X    if (i==0) return(EOF);             /* Wind up here only on EOF */
X    return(i);                         /* Handle partial buffer */
X}
X
X
X/*
X *      b u f e m p
X *
X *  Put data from an incoming packet into a file.
X */
X
Xbufemp(buffer,len)
Xchar  buffer[];                         /* Buffer */
Xint   len;                              /* Length */
X{
X    int i;                             /* Counter */
X    char t;                            /* Character holder */
X
X#ifdef EXTRADEBUG
X        if (debug > 2) printmsg("Entered bufemp");
X#endif
```

```
X    for (i=0; i<len; i++)              /* Loop thru the data field */
X    {
X        t = buffer[i];                 /* Get character */
X        if (t == MYQUOTE)              /* Control quote? */
X        {                              /* Yes */
X            t = buffer[++i];           /* Get the quoted character */
X            if ((t & 0177) != MYQUOTE)  /* Low order bits match quote char? */
X                t = ctl(t);            /* No, uncontrollify it */
X        }
X        if (t==CR && !image)           /* Don't pass CR if not in image mode */
X            continue;
X#ifdef EXTRADEBUG
X        if (debug > 2) printmsg("writing t=%c", t);
X#endif
X        putc(t,fp);
X    }
X}
X
X
X
X/*
X *  g n x t f l
X *
X *  Get next file in a file group
X */
X
Xgnxtfl()
X{
X    if (filecount-- == 0) return FALSE; /* If no more, fail */
X    if (debug) printmsg("   gnxtfl: filelist = \"%s\"\n",*filelist);
X    filnam = *(filelist++);
X    return TRUE;                   /* else succeed */
X}
X
X
X/*
X *  s p a r
X *
X *  Fill the data array with my send-init parameters
X *
X */
X
Xspar(data)
Xchar data[];
X{
X    data[0] = tochar(MAXPACKSIZ);         /* Biggest packet I can receive */
X    data[1] = tochar(reqtimint);          /* When I want to be timed out */
X    data[2] = tochar(MYPAD);           /* How much padding I need */
X    data[3] = ctl(MYPCHAR);            /* Padding character I want */
X    data[4] = tochar(MYEOL);           /* End-Of-Line character I want */
X    data[5] = MYQUOTE;                 /* Control-Quote character I send */
X}
X
```

```
X
X/*  r p a r
X *
X *  Get the other host's send-init parameters
X *
X */
X
Xrpar(data)
Xchar data[];
X{
X    spsiz = unchar(data[0]);            /* Maximum send packet size */
X    timint = unchar(data[1]);           /* When I should time out */
X    if (timint > MAXTIM) timint = MAXTIM;
X    if (timint < MINTIM) timint = MINTIM;
X    pad = unchar(data[2]);              /* Number of pads to send */
X    padchar = ctl(data[3]);             /* Padding character to send */
X    eol = unchar(data[4]);              /* EOL character I must send */
X    quote = data[5];                    /* Incoming data quote character */
X}
X
X
X/*
X *  Kermit printing routines:
X *
X *  printmsg -  like printf with "Kermit: " prepended
X *  prerrpkt - print contents of error packet received from remote host
X */
X
X
X/*
X *  p r i n t m s g
X *
X *  Print message kermit window
X */
X
X/*VARARGS1*/
Xprintmsg(fmt, a1, a2, a3, a4, a5)
Xchar *fmt;
X{
X   char *sbuf[100];
X        w_output(kermwdes,"Kermit: ");
X        sprintf(sbuf,fmt,a1,a2,a3,a4,a5);
X        w_output(kermwdes,sbuf);
X        w_output(kermwdes,"\n\r");
X}
X
X
X/*
X *  p r e r r p k t
X *
X *  Print contents of error packet received from remote host.
X */
Xprerrpkt(msg)
```

```
Xchar *msg;
X{
X    printmsg("Kermit aborting with following error from remote host:\n%s",msg);
X    return;
X}
SHAR_EOF
chmod 0600 KERMIT.C || echo "restore of KERMIT.C fails"
sed 's/^X//' << 'SHAR_EOF' > LASTCHG &&
X89 Jan 24    2.1.2
X         Removed radio button and touch-exit attributes and
X         added exit attribute to send/receive/cancel buttons
X         in kermit parameter dialog (double-click bug fix) -- BK
X88 Sep 19    2.1.1
X         Added cursor-to-bottom on font change+reshape and
X         fixed set_menu_string() bug -- BK
X88 Sep 12    Version w/ latest fixes from Brad Bosch
SHAR_EOF
chmod 0600 LASTCHG || echo "restore of LASTCHG fails"
sed 's/^X//' << 'SHAR_EOF' > LCKBLK1.ICN &&
X/* GEM Icon Definition: */
X#define lckblk1_W 0x0010
X#define lckblk1_H 0x0028
X#define lckblk1SIZE 0x0028
Xshort lckblk1[lckblk1SIZE] =
X{ 0x0005, 0x0009, 0x0001, 0x0000,
X  0x0001, 0x1FF8, 0x300C, 0x67E6,
X  0x4C32, 0x581A, 0x500A, 0xDFFB,
X  0x8001, 0x8001, 0x83C1, 0x83C1,
X  0x8181, 0x8181, 0x8181, 0x8001,
X  0xFFFF, 0x0000, 0x0FF0, 0x1818,
X  0x300C, 0x2004, 0x2004, 0x2004,
X  0x7FFE, 0x7FFE, 0x7C3E, 0x7C3E,
X  0x7E7E, 0x7E7E, 0x7E7E, 0x7FFE,
X  0x0000, 0x0000, 0x0000, 0x0000
X};
SHAR_EOF
chmod 0600 LCKBLK1.ICN || echo "restore of LCKBLK1.ICN fails"
sed 's/^X//' << 'SHAR_EOF' > MAKE.BAT &&
X# This file has not been maintained and is not supported
Xcp68 -i \dric.dir\ -i %1 %1winmain.c %1winmain.i
Xc068 %1winmain.i %1winmain.1 %1winmain.2 %1winmain.3 -f
Xrm %1winmain.i
Xc168 %1winmain.1 %1winmain.2 %1winmain.s
Xrm %1winmain.1
Xrm %1winmain.2
Xas68 -l -u %1winmain.s
Xrm %1winmain.s
Xcp68 -i \dric.dir\ -i %1 %1winsubr.c %1winsubr.i
Xc068 %1winsubr.i %1winsubr.1 %1winsubr.2 %1winsubr.3 -f
Xrm %1winsubr.i
Xc168 %1winsubr.1 %1winsubr.2 %1winsubr.s
Xrm %1winsubr.1
Xrm %1winsubr.2
```

```
Xas68 -l -u %1winsubr.s
Xrm %1winsubr.s
Xlink68 [u] %1wind.68k=gemstart,%1winsubr,
%1winmain,vdibind,gemlib,aesbind,osbind,libf
Xrelmod %1wind
Xrm %1wind.68k
SHAR_EOF
chmod 0600 MAKE.BAT || echo "restore of MAKE.BAT fails"
sed 's/^X//' << 'SHAR_EOF' > MAKEFILE &&
X# make file for uw compiled with mwc
X# GETSPEED, GETFLOW, and GETUCR use magic cookies to get the RS232
X# configuration.  Otherwise, the configuration is not known until
X# you set it.
XCFLAGS=-O -DMWC -DLOCATE='"c:\\bin"' -DGETSPEED -DGETFLOW -DGETUCR
X
Xall:    bldmform.prg mform.o win.prg startgem.prg
X    echo
X
XOBJS=   winmain.o winsubr.o mform.o resetaux.o winhash.o winio.o winproto.o\
X        winproc.o kermit.o
X
Xwin.prg: $(OBJS)
X    $(CC) $(CFLAGS) -VGEM -s -o win.prg $(OBJS)
X
Xmform.o: rmbblk1.icn lckblk1.icn
X    bldmform -ndR rmbmform rmbblk1.icn > mform.c
X    bldmform -ndR lckmform lckblk1.icn >> mform.c
X    $(CC) $(CFLAGS) -c mform.c
X
Xbldmform.prg:    bldmform.c
X    $(CC) $(CFLAGS) -o bldmform.prg bldmform.c
X
Xresetaux.o:  resetaux.s
X    $(CC) $(CFLAGS) -c resetaux.s
X
Xstartgem.prg:    startgem.s
X    as -o startgem.o startgem.s
X    ld -s -o startgem.prg startgem.o
X    rm startgem.o
SHAR_EOF
chmod 0600 MAKEFILE || echo "restore of MAKEFILE fails"
sed 's/^X//' << 'SHAR_EOF' > MFORM.C &&
X/* GEM Icon Definition: */
X#define rmbblk1_W 0x0010
X#define rmbblk1_H 0x0028
X#define rmbblk1SIZE 0x0028
Xshort rmbmform[rmbblk1SIZE] =
X{ 0x000A, 0x0003, 0x0001, 0x0000,
X  0x0001, 0xFFFF, 0x8037, 0xB82F,
X  0xB801, 0xB82F, 0xB837, 0x803F,
X  0xBFA0, 0xBFA0, 0xBFA0, 0xBBA0,
X  0xB1A0, 0xAAA0, 0xBFA0, 0x8020,
X  0xFFE0, 0x0000, 0x7FC8, 0x47D0,
```

```
X  0x47FE, 0x47D0, 0x47C8, 0x7FC0,
X  0x4040, 0x4040, 0x4040, 0x4440,
X  0x4E40, 0x5540, 0x4040, 0x7FC0,
X  0x0000, 0x0000, 0x0000, 0x0000
X};
X/* GEM Icon Definition: */
X#define lckblk1_W 0x0010
X#define lckblk1_H 0x0028
X#define lckblk1SIZE 0x0028
Xshort lckmform[lckblk1SIZE] =
X{ 0x0005, 0x0009, 0x0001, 0x0000,
X  0x0001, 0x1FF8, 0x300C, 0x67E6,
X  0x4C32, 0x581A, 0x500A, 0xDFFB,
X  0x8001, 0x8001, 0x83C1, 0x83C1,
X  0x8181, 0x8181, 0x8181, 0x8001,
X  0xFFFF, 0x0000, 0x0FF0, 0x1818,
X  0x300C, 0x2004, 0x2004, 0x2004,
X  0x7FFE, 0x7FFE, 0x7C3E, 0x7C3E,
X  0x7E7E, 0x7E7E, 0x7E7E, 0x7FFE,
X  0x0000, 0x0000, 0x0000, 0x0000
X};
SHAR_EOF
chmod 0600 MFORM.C || echo "restore of MFORM.C fails"
sed 's/^X//' << 'SHAR_EOF' > README &&
X
X        UW (Unix (tm) Windows) for the Atari ST
X
XThis program was written by Hans-Martin Mosner at the University of
XDortmund in West Germany.  It has since been enhanced by Brad Bosch
Xand Brian Katzung at Lachman Associates, Inc., in Naperville, Illinois.
X
XNote that this disk represents a beta test version, and is provided on
Xa strictly as-is basis, and we assume no responsibility or liability
Xfor any aspect of this software.
X
XThis version of uw provides only adm31 emulation and the most basic
XUW protocol.  The adm31 emulation has been extended slightly.  See
Xthe end of the Features section for a description of the extensions.
X
XFeatures added by Brad Bosch and Brian Katzung:
X
X    o Jump scrolling occurs if screen updates can't keep up with
X      transmission from the host.
X
X    o The input window may be different than the top window.
X
X    o A printer copy of output to each window may be enabled or
X      disabled on a window-by-window basis in various flavors.
X
X    o <Shift><Help> sends a break.
X
X    o There is a menu item to reset flow control if the Atari and
X      the host get out of synchronization.
```

X
X   o Fonts "windalt.fnt", "windoth.fnt", and "windtny.fnt" are
X   supported in addition to "windstd.fnt". The "windtny" font is
X   small enough to allow four 24 x 80 windows on the screen without
X   overlapping.
X
X   o The smaller system font is offered in addition to the larger
X   one. Some of the characters in the smaller font are "improved"
X   to increase readability.
X
X   o The user may now select whether or not a window will be
X   "topped" when a ^G (BEL) is output to it.
X
X   o The user may choose to have ^G (BEL) flash the screen, ring
X   the bell, neither, or both.
X
X   o When not selecting an input window, pressing the right mouse
X   button on the menu bar toggles the menu bar sensitivity.
X
X   o The close-box works even when the UW protocol is not running.
X   This is useful for changing the font size or other options.
X
X   o The RTS (request to send) and DTR (data terminal ready) lines
X   can be cleared and asserted.
X
X   o Function keys can be used and defined up to 40 characters each.
X
X   o Function keys and other global option values can be saved and
X   loaded. A file called "win.cnf", if present, is read at startup
X   time.
X
X   o The right mouse button can be used to copy text from the screen
X   and send it to a window as keyboard input. Click twice on a
X   window border to send copied text. Click once on the boarder to
X   select the window for input. Select text by dragging with the
X   right button pressed. Inside a border, a single right button
X   click selects a word and a double click selects a line.
X
X   o Emulation window size can now be selected when a window is opened.
X   The fixed sizes available in the menu can be changed with a resource
X   editor without recompiling. Default sizes are font dependent.
X
X   o A window menu item was added to allow the bottom window to be sent
X   to the top even if it is competely hidden. Another item sends the
X   top window to the bottom. The latter is somewhat slow due to
X   shortcomings in GEM.
X
X   o The main keyboard digit keys when used while the alt key is down
X   will select that window number to be the top and keyboard input
X   window. The first window is alt-1, the last is alt-7. Windows
X   are numbered as gem numbers them. Alt-key bindings for lower
X   case letters are scanned from the menu strings in the resource
X   file. These bindings are indicated by the ^E (close box)

```
X      character.
X
X    o A window menu item was added to shrink the current window to an
X      iconic size.  The second-from-the-top non-iconic window becomes
X      the current window.  The fuller box is used to reverse the process.
X
X    o An option is now available to cause new windows to be created
X      without the sliders and arrows. The title bar (and close and
X      fuller boxes) may also be eliminated by another option.  Menu
X      items have been added to support most of the functionality lost
X      when the window controls are turned off.
X
X    o Additional help information was added to the about dialog.
X
X    o Startgem is now included in the distribution.  This program, if
X      placed in the auto folder, will attempt to start a:\bin\uw.prg
X      automatically when the ST is booted up.
X
X    o A variable called "locpath" contains the name of a directory to
X      scan after the current directory when looking for configuration
X      and font files.  The string is 64 bytes long and defaults to
X      c:\bin.
X
X    o An RS232 configuration dialog has been added.  There is no pro-
X      vision for putting this information in the configuration file.
X      Bug view:  There is no auto-configuration.  Feature view:  Changes
X      persist across executions.
X
X    o One can now capture host output to a file on a per-window basis.
X      Capture mode is terminated by setting an empty file name.
X
X    o Added keyboard lock to file menu.  This locks the keyboard and mouse
X      until your password is reentered.
X
X    o Added change working directory item to file menu.  This is usefull
X      for kermit and the exec command entry.
X
X    o Added execute command entry.  This allows uw to be suspended while
X      another command is executed.  Note that mwc startup for gem does not
X      preserve the environment variables.  The default command and
X      parameters are part of the config file.
X
X    o Add basic kermit functionality.  Kermit can operate under the uw
X      protocol in a window concurent with interactive terminal windows.
X      Sending multiple files is not yet supported.
X
X
XExtensions to Adm31 emulation:
X
X    o If the host sends a window the sequence ESC [18t, the window will
X      respond with ESC [8;<rows>;<cols>t, where <rows> and <cols> specify
X      the window's size in characters.
X
```

```
X    o The insert line and delete line escape sequences (ESC E and ESC R,
X      respectively) allow a repeat count (ie, ESC 5E inserts 5 lines).
X
XBugs fixed:
X
X    o Exiting in "fast" mode used to leave a window open.  The
X      system would crash on the next run unless you had returned
X      to the desktop.
X
X    o Actions on windows now check that there are windows to act upon.
X
X
XKnown bugs:
X
X    o Because of the extra memory used in the jump scrolling
X      algorithm, the bottom row of pixels of an earlier line
X      sometimes appears immediately below the title bar.
X
X    o GEM doesn't let you completely eliminate the menu bar and
X      increase your workspace.
X
X    o Desk accessories sometimes leave damage on the screen.
X
X    o GEM doesn't send "top" messages to the top window, so you
X      can't use the left mouse button as a short hand to select
X      the top window for input.  Use the right button on the window
X      title bar instead.
X
X    o The rsconf() baud rates listed in the manual as 75 and 50 baud
X      really result in system timer settings for 120 and 80 baud.
X      The RS232 configuration dialog allows you to set these two baud
X      rates, but decodes them as 300 or 200 baud, respectively.
X
X    o The RS232 configuration code uses some trickery to determine the
X      current port configuration.  Some parts of this code may need to
X      be turned off or adapted to run on other hardware or if system
X      calls stop returning undocumented values.
SHAR_EOF
chmod 0600 README || echo "restore of README fails"
sed 's/^X//' << 'SHAR_EOF' > RESETAUX.S &&
X/
X/ Reset the Aux rs232 port.  Usage:
X/   char *rs232_iorec = Iorec(0);    /* Do this first   */
X/
X/   if (rs232_reset())
X/       Cauxout(0x11);          /* sent an XOFF        */
X/
X/ This subroutine was written by Martin Minow, Arlington, MA
X/ and is in the public domain.
X/
X/   .shrd                   / Public data
X/   .globl    rs232_iorec_       / -> aux io record. see below
X    .shri                   / Public instructions
```

```
X    .globl   rs232_reset_
X
Xrs232_reset_:
X    move.l   d2,-(sp)      / Save a scratch register
X    clr.l    -(sp)             / Super(0L) switches to super mode
X    move.w   $0x20,-(sp)       / Super opcode
X    trap $1           / Call TOS
X    addq.l   $6,sp             / Clear stack
X    move.l   d0,-(sp)      / Save old stack for now.
X/
X/ If you don't already have the iorec as a readily-available global,
X/ uncomment the following code (which is, of course, untested)
X/
X    clr.w    -(sp)             / Aux port is device zero
X    move.w   $14,-(sp)     / Iorec
X    trap $14          / Xbios(14)
X    addq.l   $4,sp             / Clear stack
X/ End of "get iorec" code. We already have this information:
X/   movea    rs232_iorec_,a0        / a0 -> I/O record
X    movea    d0,a0             / a0 -> I/O record
X    move sr,-(sp)      / Save cpu status
X    ori  $0x700,sr     / Disable interrupts
X/   clr.l    6(a0)             / Clear input  head, tail -- crashes!
X    clr.l    20(a0)            / Clear output head, tail
X    clr.l    d2            / Clear result
X    or.b 30(a0),d2     / Get old rcv xoff state
X    clr.w    30(a0)            / Cancel rcv, xmt xoff states
X    move (sp)+,sr      / Enable interrupts
X    move.w   $0x20,-(sp)       / Super -- old_stack is still at (sp)
X    trap $1           / Exit super mode
X    addq $6,sp             / Clean junk from stack
X    move.l   d2,d0             / Get result
X    move.l   (sp)+,d2      / Restore d2
X    rts               / Exit routine
SHAR_EOF
chmod 0600 RESETAUX.S || echo "restore of RESETAUX.S fails"
sed 's/^X//' << 'SHAR_EOF' > RMBBLK1.ICN &&
X/* GEM Icon Definition: */
X#define rmbblk1_W 0x0010
X#define rmbblk1_H 0x0028
X#define rmbblk1SIZE 0x0028
Xshort rmbblk1[rmbblk1SIZE] =
X{ 0x000A, 0x0003, 0x0001, 0x0000,
X  0x0001, 0xFFFF, 0x8037, 0xB82F,
X  0xB801, 0xB82F, 0xB837, 0x803F,
X  0xBFA0, 0xBFA0, 0xBFA0, 0xBBA0,
X  0xB1A0, 0xAAA0, 0xBFA0, 0x8020,
X  0xFFE0, 0x0000, 0x7FC8, 0x47D0,
X  0x47FE, 0x47D0, 0x47C8, 0x7FC0,
X  0x4040, 0x4040, 0x4040, 0x4440,
X  0x4E40, 0x5540, 0x4040, 0x7FC0,
X  0x0000, 0x0000, 0x0000, 0x0000
X};
```

```
SHAR_EOF
chmod 0600 RMBBLK1.ICN || echo "restore of RMBBLK1.ICN fails"
sed 's/^X//' << 'SHAR_EOF' > STARTGEM.S &&
X/
X/   STARTGEM.S    6-MAR-1987    by GHOST-Soft
X/
X/   This program is used to start GEM programs from within the AUTO folder.
X/
X/   The code is probably highly nonportable, as it heavily depends on the
X/   internal organisation of code in TOS/GEM.
X/   As this program works with ROM- as well as all disk-based versions
X/   of German TOS, I'm pretty sure that this program works with foreign
X/   versions of TOS, too.
X/   All addresses given in square brackets apply to the German ROM-TOS.
X/
X/   Assemble with Mark Williams assembler:
X/        as -o startgem.o startgem.s
X/        gemfix startgem.o startgem.prg
X/        strip startgem.prg
X/             _or_
X/        as -o startgem.o startgem.s
X/        ld -o startgem.prg -s startgem.o
X/
X
Xlinef=       0x2c
Xvbllist1=    0x4d2
X
Xgemdos=      1
X  cconws=    0x9
X  fsetdta=   0x1a
X  super= 0x20
X  ptermres=  0x31
X  fsfirst=   0x4e
Xxbios=       14
X  vsync= 0x25
X
X    .shri
Xstart:
X    lea  basepage(pc),a0
X    move.l   4(a7),(a0)    / get basepage address
X
X    pea  msg(pc)
X    move.w   $cconws,-(a7) / print line
X    trap $gemdos
X    addq.w   $6,a7
X
X    moveq    $70,d2        / wait 1 second = 71 vblanks
X1:  move.w   $vsync,-(a7)  / wait for vertical blank
X    trap $xbios
X    addq.w   $2,a7
X    dbra d2,1b
X
X    clr.l    -(a7)
```

```
X    move.w    $super,-(a7)  / enter super mode
X    trap $gemdos
X    addq.w    $6,a7         / keep old super sp in d0
X
X    moveq     $-1,d2
X    move.l    d2,linef / set line-f emulator trap
X    lea  vblank(pc),a2
X    move.l    a2,vbllist1   / set vblank vector #1  {0..7}
X
X    move.l    d0,-(a7)
X    move.w    $super,-(a7)  / exit super mode
X    trap $gemdos
X    addq.w    $6,a7
X
X    lea  basepage(pc),a0    / compute number of bytes to keep
X    suba.l    basepage(pc),a0
X    clr.w     -(a7)         / 0 = no error
X    move.l    a0,-(a7)
X    move.w    $ptermres,-(a7)    / terminate and stay resident
X    trap $gemdos       / this call never returns
X
Xvblank:
X    moveq     $-1,d0
X    cmp.l     linef,d0
X    beq.s     1f
X    clr.l     vbllist1 / clear vblank vector #1
X    movea.l   linef,a0
X    lea  old_lf(pc),a1
X    move.l    a0,(a1)       / save old line-f vector (usually in low RAM)
X    lea  new_lf(pc),a1
X    move.l    a1,linef / install new one
X
X    / find the address of the buffer that GEM uses for the
X    / name of the program that it is going to execute
X    movea.l   0x16(a0),a0   / get start of line-f table from instruction
X    / [a0 = fee8bc]         move.l $0xfee8bc,a0
X    movea.l   0x7c8(a0),a0  / get address of routine 'f7c8' from table
X    / [a0 = fd8fc4]
X    movea.l   0xa(a0),a2    / get address from instruction
X    / [a2 = 73e4]      move.l $0x73e4,a5
X    adda.w    0x10(a0),a2   / add offset 0x1f56 from instruction
X    / [a2 = 933a]      lea 0x1f56,a0
X    lea  pointer(pc),a1
X    move.l    a2,(a1)       / save buffer address
X1:  rts
X
Xnew_lf:
X    movea.l   2(a7),a0 / get addr. of instr. that caused exception
X    cmpi.w    $0xf08c,(a0)  / 'f08c' is the line-f instr. we're looking for
X    bne.w     exit
X    move.l    old_lf(pc),linef   / restore old line-f vector
X
X    movea.l   pointer(pc),a0
```

```
X    lea  filename(pc),a1
X1:  move.b   (a1)+,(a0)+   / copy filename into GEM buffer space
X    bne.s    1b
X
X    pea  dta
X    move.w   $fsetdta,-(a7)     / set DTA
X    trap $gemdos
X    addq.l   $6,a7
X    move.w   $0x27,-(a7)   / don't look at volumes and directories
X    pea  filename(pc)
X    move.w   $fsfirst,-(a7)     / search first
X    trap $gemdos
X    addq.l   $8,a7
X    tst  d0       / status != 0 indicates error
X    bne.s    exit
X
X    movea.l  2(a7),a0 / get addr. of instr. that caused exception
X    / [a0 = fed188]
X2:  cmpi.w   $0xf4b8,(a0)  / look for line-f instruction 'f4b8'
X    beq.s    doload
X    addq.l   $2,a0
X    cmpa.l   $0xfefffe,a0  / test for ROM end
X    beq.s    exit
X    cmpa.l   $0x078000,a0  / test for RAM end (on 512K machine)
X    bne.s    2b       / {RAM-TOS usually loads much lower}
X
Xexit:
X    movea.l  old_lf(pc),a0 / resume execution of regular line-f handler
X    jmp  (a0)
X
Xdoload:
X    / [a0 = fe1de2]
X    move.w   (a7)+,d2  / pop status word
X    addq.w   $4,a7         / destroy addr. of instr. 'f08c'
X    move.w   d2,sr         / restore status register
X    adda.w   $56,a7        / destroy parameters of instr. 'f08c'
X    jmp  (a0)     / jump to instruction 'f4b8'
X
X    .prvd
Xdta:
X    .blkb    44
Xold_lf:
X    .blkl    1
Xpointer:
X    .blkl    1
X
Xmsg:
X    .ascii   "\033\105Autostart of "
Xfilename:
X    .ascii   "a:\\bin\\UW.PRG"
Xnulls:
X    .blkb    20       / these nulls allow easy patching of code file
X
```

```
X    .even
Xbasepage:
X    .ascii   "EOPA"        / magic number indicates End Of Patch Area
SHAR_EOF
chmod 0600 STARTGEM.S || echo "restore of STARTGEM.S fails"
sed 's/^X//' << 'SHAR_EOF' > UW.H &&
X/*
X *  uw command bytes
X *
X * Copyright 1985 by John D. Bruner.  All rights reserved.  Permission to
X * copy this program is given provided that the copy is not sold and that
X * this copyright notice is included.
X *
X *
X * Two types of information are exchanged through the 7-bit serial line:
X * ordinary data and command bytes.  Command bytes are preceeded by
X * an IAC byte.  IAC bytes and literal XON/XOFF characters (those which
X * are not used for flow control) are sent by a CB_FN_CTLCH command.
X * Characters with the eighth bit set (the "meta" bit) are prefixed with
X * a CB_FN_META function.
X *
X * The next most-significant bit in the byte specifies the sender and
X * recipient of the command.  If this bit is clear (0), the command byte
X * was sent from the host computer to the Macintosh; if it is set (1)
X * the command byte was sent from the Macintosh to the host computer.
X * This prevents confusion in the event that the host computer
X * (incorrectly) echos a command back to the Macintosh.
X *
X * The remaining six bits are partitioned into two fields.  The low-order
X * three bits specify a window number from 1-7 (window 0 is reserved for
X * other uses) or another type of command-dependent parameter.  The next
X * three bits specify the operation to be performed by the recipient of
X * the command byte.
X *
X * Note that the choice of command bytes prevents the ASCII XON (021) and
X * XOFF (023) characters from being sent as commands.  CB_FN_ISELW commands
X * are only sent by the Macintosh (and thus are tagged with the CB_DIR_MTOH
X * bit).  Since XON and XOFF data characters are handled via CB_FN_CTLCH,
X * this allows them to be used for flow control purposes.
X */
X
X#define  IAC      0001     /* interpret as command */
X#define  CB_DIR        0100     /* command direction: */
X#define  CB_DIR_HTOM   0000     /*   from host to Mac */
X#define  CB_DIR_MTOH   0100     /*   from Mac to host */
X#define  CB_FN         0070     /* function code: */
X#define  CB_FN_NEWW    0000     /*   new window */
X#define  CB_FN_KILLW   0010     /*   kill (delete) window */
X#define  CB_FN_ISELW   0020     /*   select window for input */
X#define  CB_FN_OSELW   0030     /*   select window for output */
X#define  CB_FN_META    0050     /*   add meta to next data char */
X#define  CB_FN_CTLCH   0060     /*   low 3 bits specify char */
X#define  CB_FN_MAINT   0070     /*   maintenance functions */
```

```
X#define  CB_WINDOW 0007     /* window number mask */
X#define  CB_CC         0007     /* control character specifier: */
X#define  CB_CC_IAC 1        /*   IAC */
X#define  CB_CC_ON  2        /*   XON */
X#define  CB_CC_OFF 3        /*   XOFF */
X#define  CB_MF         0007     /* maintenance functions: */
X#define  CB_MF_ENTRY   0        /*   beginning execution */
X#define  CB_MF_EXIT    7        /*   execution terminating */
X#define  NWINDOW       7        /* maximum number of windows */
X
X/* CB_CC_XON and CB_CC_XOFF give the same name on the DR C compiler.
X * Changed them to CB_CC_ON and CB_CC_OFF
X * hmm@unido
X */
SHAR_EOF
chmod 0600 UW.H || echo "restore of UW.H fails"
sed 's/^X//' << 'SHAR_EOF' > VERSION &&
XUW Version 2.1.2  24 Jan 1989
SHAR_EOF
chmod 0655 VERSION || echo "restore of VERSION fails"
sed 's/^X//' << 'SHAR_EOF' > WIN-.CNF &&
Xaudibell=1
Xvisibell=1
Xtopbell=0
Xfast
Xnooverstrike
Xnosliders
Xtitles
Xbufsiz=256
Xcmdpath=E:\BIN\*.*
Xcmdname=MSH.PRG
Xcmdargs=c:\src\uw\profile
Xfont=3
Xf1=at^m
Xf2=atdt5559100,,,,410,,^m
Xf3=atdt5551205^m
Xf4=atdt5551102^m
Xf5=ata^m
Xf6=ATDT5551272^m
Xf7=^A6^M
Xf8=^A7^M
Xf9=^A8^M
Xf10=ati4^m
Xf11=^A10^M
Xf12=^A11^M
Xf13=^A12^M
Xf14=^A13^M
Xf15=^A14^M
Xf16=^A15^M
Xf17=^A16^M
Xf18=^A17^M
Xf19=^A18^M
Xf20=^A19^M
```

```
Xf21=^A20^M
Xf22=^A21^M
Xf23=^A22^M
Xf24=^A23^M
Xf25=^A24^M
Xf26=^A25^M
Xf27=^A26^M
Xf28=^A27^M
Xf29=^A28^M
Xf30=^A29^M
Xf31=^A30^M
Xf32=^A31^M
Xf33=^A32^M
Xf34=^A33^M
Xf35=^A34^M
Xf36=^A35^M
Xf37=^A36^M
Xf38=^A37^M
Xf39=^A38^M
Xf40=^A39^M
SHAR_EOF
chmod 0655 WIN-.CNF || echo "restore of WIN-.CNF fails"
sed 's/^X//' << 'SHAR_EOF' > WIN.CNF &&
Xaudibell=1
Xvisibell=1
Xtopbell=0
Xfast
Xnooverstrike
Xnosliders
Xnotitles
Xbufsiz=256
Xcmdpath=E:\BIN\*.*
Xcmdname=MSH.PRG
Xcmdargs=c:\src\uw\profile
Xfont=2
Xf1=atx6d5559100,,,,,410,,,^M
Xf2=atx6d5559100,,,,,420,,,^M
Xf3=atx6d18005559848^M
Xf4=^A3^M
Xf5=atx6d14155554619^M
Xf6=atx6d14155556755^M
Xf7=atx6d14155554127^M
Xf8=atdt*70;^M
Xf9=^A8^M
Xf10=adm31^M
Xf11=^A10^M
Xf12=^A11^M
Xf13=^A12^M
Xf14=^A13^M
Xf15=^A14^M
Xf16=^A15^M
Xf17=^A16^M
Xf18=^A17^M
```

```
Xf19=^A18^M
Xf20=^A19^M
Xf21=^A20^M
Xf22=^A21^M
Xf23=^A22^M
Xf24=^A23^M
Xf25=^A24^M
Xf26=^A25^M
Xf27=^A26^M
Xf28=^A27^M
Xf29=^A28^M
Xf30=^A29^M
Xf31=^A30^M
Xf32=^A31^M
Xf33=^A32^M
Xf34=^A33^M
Xf35=^A34^M
Xf36=^A35^M
Xf37=^A36^M
Xf38=^A37^M
Xf39=^A38^M
Xf40=^A39^M
SHAR_EOF
chmod 0655 WIN.CNF || echo "restore of WIN.CNF fails"
sed 's/^X//' << 'SHAR_EOF' > WIND.H &&
X#define MENUBAR 0     /* TREE */
X#define ABOUT 1   /* TREE */
X#define NEWNAME 2     /* TREE */
X#define DABOUT 12     /* OBJECT in TREE #0 */
X#define FQUITUW 21    /* OBJECT in TREE #0 */
X#define FQUIT 26      /* OBJECT in TREE #0 */
X#define MFAST 33      /* OBJECT in TREE #0 */
X#define WRENAME 47    /* OBJECT in TREE #0 */
X#define FNTOWN 69     /* OBJECT in TREE #0 */
X#define FNTSYS 68     /* OBJECT in TREE #0 */
X#define BUTTON 5      /* OBJECT in TREE #1 */
X#define FLD1 1    /* OBJECT in TREE #2 */
X#define OKRENAME 3    /* OBJECT in TREE #2 */
X#define FNTALT 70     /* OBJECT in TREE #0 */
X#define DESK 3    /* OBJECT in TREE #0 */
X#define PRTBOTOM 74   /* OBJECT in TREE #0 */
X#define PRTSTOP 77    /* OBJECT in TREE #0 */
X#define FUNCTKEY 36   /* OBJECT in TREE #0 */
X#define PRTTOP 75     /* OBJECT in TREE #0 */
X#define RESETAUX 43   /* OBJECT in TREE #0 */
X#define AUDIBELL 37   /* OBJECT in TREE #0 */
X#define VISIBELL 38   /* OBJECT in TREE #0 */
X#define TOPONBEL 39   /* OBJECT in TREE #0 */
X#define ASSRD 40      /* OBJECT in TREE #0 */
X#define RESRD 41      /* OBJECT in TREE #0 */
X#define PRTWIND 76    /* OBJECT in TREE #0 */
X#define FUNCTEDT 3    /* TREE */
X#define FUNCEXIT 7    /* OBJECT in TREE #3 */
```

```
X#define FUNCSHOW 6     /* OBJECT in TREE #3 */
X#define FUNCPREV 5     /* OBJECT in TREE #3 */
X#define FUNCNEXT 4     /* OBJECT in TREE #3 */
X#define FUNCNAME 10    /* OBJECT in TREE #3 */
X#define FUNCBODY 12    /* OBJECT in TREE #3 */
X#define FUNCENT 16     /* OBJECT in TREE #3 */
X#define LOADCONF 23    /* OBJECT in TREE #0 */
X#define SAVECONF 24    /* OBJECT in TREE #0 */
X#define WINDSIZE 4     /* TREE */
X#define WINDROWS 7     /* OBJECT in TREE #4 */
X#define WINDCOLS 8     /* OBJECT in TREE #4 */
X#define WINDCANC 5     /* OBJECT in TREE #4 */
X#define WINDOK 6       /* OBJECT in TREE #4 */
X#define SHRNKWIN 48    /* OBJECT in TREE #0 */
X#define BOTTOMTO 50    /* OBJECT in TREE #0 */
X#define INPUTWIN 54    /* OBJECT in TREE #0 */
X#define HIDEWIN 49     /* OBJECT in TREE #0 */
X#define WINSTYLE 34    /* OBJECT in TREE #0 */
X#define MOREINF1 6     /* OBJECT in TREE #1 */
X#define INFO1 5   /* TREE */
X#define CONT1 4   /* OBJECT in TREE #5 */
X#define INFO2 6   /* TREE */
X#define CONT2 8   /* OBJECT in TREE #6 */
X#define INFO3 7   /* TREE */
X#define INFO4 8   /* TREE */
X#define INFO5 9   /* TREE */
X#define CONT3 2   /* OBJECT in TREE #7 */
X#define CONT4 2   /* OBJECT in TREE #8 */
X#define CONT5 2   /* OBJECT in TREE #9 */
X#define FNTBIG 67      /* OBJECT in TREE #0 */
X#define FNTTINY 71     /* OBJECT in TREE #0 */
X#define FNTUNKNW 72    /* OBJECT in TREE #0 */
X#define OSIZEWIN 51    /* OBJECT in TREE #0 */
X#define MOVEWIN 52     /* OBJECT in TREE #0 */
X#define CLOSEWIN 53    /* OBJECT in TREE #0 */
X#define WINTITLE 35    /* OBJECT in TREE #0 */
X#define PASTE 55       /* OBJECT in TREE #0 */
X#define CAPTURE 25     /* OBJECT in TREE #0 */
X#define RSCONF 10      /* TREE */
X#define BB1920 7       /* OBJECT in TREE #10 */
X#define BB960 8   /* OBJECT in TREE #10 */
X#define BB480 9   /* OBJECT in TREE #10 */
X#define BB360 10       /* OBJECT in TREE #10 */
X#define BB240 11       /* OBJECT in TREE #10 */
X#define BB200 12       /* OBJECT in TREE #10 */
X#define BB180 13       /* OBJECT in TREE #10 */
X#define BB120 14       /* OBJECT in TREE #10 */
X#define BB60 15   /* OBJECT in TREE #10 */
X#define BB30 16   /* OBJECT in TREE #10 */
X#define BB20 17   /* OBJECT in TREE #10 */
X#define BB15 18   /* OBJECT in TREE #10 */
X#define BB13 19   /* OBJECT in TREE #10 */
X#define BB11 20   /* OBJECT in TREE #10 */
```

```
X#define BB7 21    /* OBJECT in TREE #10 */
X#define BB5 22    /* OBJECT in TREE #10 */
X#define BCB5 24   /* OBJECT in TREE #10 */
X#define BCB6 25   /* OBJECT in TREE #10 */
X#define BCB7 26   /* OBJECT in TREE #10 */
X#define BCB8 27   /* OBJECT in TREE #10 */
X#define PBNONE 30     /* OBJECT in TREE #10 */
X#define PBEVEN 31     /* OBJECT in TREE #10 */
X#define PBODD 32      /* OBJECT in TREE #10 */
X#define FCBNONE 40    /* OBJECT in TREE #10 */
X#define FCBXON 41     /* OBJECT in TREE #10 */
X#define FCBRTS 42     /* OBJECT in TREE #10 */
X#define RCCAN 1   /* OBJECT in TREE #10 */
X#define RCOK 2    /* OBJECT in TREE #10 */
X#define SBB1 36   /* OBJECT in TREE #10 */
X#define SBB15 37      /* OBJECT in TREE #10 */
X#define SBB2 38   /* OBJECT in TREE #10 */
X#define SETCONF 42    /* OBJECT in TREE #0 */
X#define BUFSIZ1 44    /* OBJECT in TREE #10 */
X#define BUFSIZ2 45    /* OBJECT in TREE #10 */
X#define BUFSIZ32 46   /* OBJECT in TREE #10 */
X#define LOCKINFO 11   /* TREE */
X#define FLOCKKEY 27   /* OBJECT in TREE #0 */
X#define COMMAND 28    /* OBJECT in TREE #0 */
X#define SETPATH 29    /* OBJECT in TREE #0 */
X#define PARAM 12      /* TREE */
X#define PARAMSTR 3    /* OBJECT in TREE #12 */
X#define OKEXEC 1      /* OBJECT in TREE #12 */
X#define KERMIT 30     /* OBJECT in TREE #0 */
X#define KERMPARM 13   /* TREE */
X#define DEBUGENA 10   /* OBJECT in TREE #13 */
X#define DEBUG1 11     /* OBJECT in TREE #13 */
X#define DEBUG2 13     /* OBJECT in TREE #13 */
X#define DEBUG3 12     /* OBJECT in TREE #13 */
X#define IMAGEMOD 7    /* OBJECT in TREE #13 */
X#define ASCIIMOD 8    /* OBJECT in TREE #13 */
X#define CONVNAME 9    /* OBJECT in TREE #13 */
X#define SENDFILE 2    /* OBJECT in TREE #13 */
X#define RECFILE 3     /* OBJECT in TREE #13 */
X#define KERMEXIT 4    /* OBJECT in TREE #13 */
X#define REMSHELL 59   /* OBJECT in TREE #0 */
X#define SHELL32 60    /* OBJECT in TREE #0 */
X#define SHELL24 61    /* OBJECT in TREE #0 */
X#define SHELL12 62    /* OBJECT in TREE #0 */
X#define SHELL8 63     /* OBJECT in TREE #0 */
X#define SHELLOTH 64   /* OBJECT in TREE #0 */
X#define WINRESIZ 57   /* OBJECT in TREE #0 */
X#define WINCLEAR 56   /* OBJECT in TREE #0 */
X#define LOCCOPY 65    /* OBJECT in TREE #0 */
X#define OVERSTRI 45   /* OBJECT in TREE #0 */
X#define MFREE 31      /* OBJECT in TREE #0 */
SHAR_EOF
chmod 0600 WIND.H || echo "restore of WIND.H fails"
```

```
sed 's/^X//' << 'SHAR_EOF' > WINDEFS.H &&
X/* header file with various definitions
X */
X
X/* this one should really be in stdio.h ...
X */
X
X#ifndef NULL
X#define NULL ((char *) 0)
X#endif
X
X#define TRUE 1
X#define FALSE 0
X
X/* constants
X */
X
X#define ALTINDICATOR '\05'
X#define MAXFUNCLEN 41  /* Maximum length of function key string */
X#define BREAK_TICKS 20 /* length of break in clock() ticks (5ms * 20 = .1s) */
X#define MAXSCROLLED 8
X#define MAX_WIND 8
X#define NFSTRINGS 42            /* number of function keys */
X#define WI_WITHSLD (SIZER|UPARROW|DNARROW|VSLIDE|LFARROW|RTARROW|HSLIDE)
X#define WI_NOSLD (NAME|MOVER|CLOSER|FULLER)
X
X
X#define WI_ROWS 80
X#define WI_LINES 24
X#define WI_WCHAR 8
X#define WI_HCHAR 16
X#define WI_WPIX (WI_ROWS*WI_WCHAR+16)
X#define WI_WWORDS ((WI_WPIX+15)/16)
X#define WI_HPIX (WI_LINES*WI_HCHAR)
X#define WI_SIZE ((long)WI_WWORDS*WI_HPIX*2)
X#define XON '\021'
X#define XOFF '\023'
X
X#define xmitcmd(x)    { Bconout(1, IAC); Bconout(1, CB_DIR_MTOH|x); }
X
X#define MINMAX(x, y, z) ( x<y ? y : (x>z ? z :x) )
X
X#define HASHBITS 9         /* number of bits of hash for character lookup*/
X
X#define FM_INVERT 12       /* logic parameter to copy_fm */
X#define FM_COPY    3       /* copy source to dest */
X
X#define X0 1
X#define Y0 1
X
X/* emulation FSM states
X */
X
```

```
X#define S_NORMAL  0
X#define S_ESC         1
X#define S_ESC1        2
X#define S_ESC2        3
X#define S_ESC3        4
X#define S_STATUS  5
X#define S_ESCA        6
X
X#define min(a,b) ( a < b ? a : b )
X#define max(a,b) ( a > b ? a : b )
X/* data types used
X */
X
Xtypedef struct mfdb {
X  short *ptr;              /* ptr to memory holding the image */
X  short wpix;              /* width in pixels */
X  short hpix;              /* height in pixels */
X  short wwords;            /* width in words */
X  short format;            /* 0 for machine-dependent */
X  short planes;            /* 1 for hires screen */
X} MFDB;
X
Xtypedef struct {
X  unsigned char h_t[1<<HASHBITS];   /* hash table */
X  struct {
X    char h_try;                /* try this character next */
X    char h_next;           /* index of next entry in table */
X  } h_colision[128];           /* colision lists formed here */
X} HTBL;
X
Xtypedef struct fnt {
X  int inc_x, inc_y;        /* real size of one char */
X  int def_win_x, def_win_y;     /* default window size in chars for this font */
X  HTBL f_hash;             /* hash table for this font */
X  char f_data[2048];       /* character data */
SHAR_EOF
echo "End of part 2, continue with part 3"
echo "3" > s2_seq_.tmp
exit 0


From 3750501919125690997
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,34b04142a5cc1c70,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-02 14:06:46 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!csd4.milw.wisc.edu!bionet!agate!
saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i021: uw -- Unix Windows windowing terminal software  part03/05
Keywords: shar
Message-ID: <6553@saturn.ucsc.edu>
```

Date: 2 Mar 89 22:06:46 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1738
Approved: koreth@ssyx.ucsc.edu

Submitted-by: sun.com!laidbak!katzung (Brian Katzung)
Posting-number: Volume 2, Issue 21
Archive-name: uw/part03

```
#!/bin/sh
# this is part 3 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file WINDEFS.H continued
#
CurArch=3
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> WINDEFS.H
X} FNT;
X
Xtypedef struct wi_str {
X  int port;           /* the UW port identifier */
X  int fulled,used;        /* flags */
X  int inverse;            /* inverse chars */
X  int insmode;            /* insert mode on */
X  int curstate;           /* state of the cursor */
X  int x, y, w, h;     /* work area on screen */
X  int px, py, pw, ph;     /* previous location of entire window */
X  int x_off, y_off;       /* offset for visible part */
X  int px_off, py_off;     /* previous offset */
X  int m_off;          /* memory offset */
X  int wi_w, wi_h;     /* max w,h */
X  int cur_x, cur_y;       /* current cursor position */
X  int top_y;          /* position of top of circular window buffer */
X  int state;          /* state for the emulation FSM */
X  FNT *font;          /* the font in use */
X  MFDB wi_mf;             /* memory form for this window */
X  char name[80];      /* name of this window */
X  char nuname[80];        /* new name under construction */
X  char dname[80];     /* name actually displayed (w/ flags, etc.) */
X                 /* (dname is set in w_rename) */
X  int nuptr;          /* curr. len of new name or parameter assembly*/
X  int ptr_status;     /* status of print loging for window */
X  int x_chrs, y_chrs;     /* size of terminal emulation */
X  int top_age;            /* number of windows topped after this window */
X  int wi_style;           /* style of window (see WI_*SLD) */
```

```
X  char wi_fpath[80];       /* log file path */
X  char wi_fname[20];       /* log file name */
X  FILE *wi_lfd;            /* log file descriptor */
X  int kerm_act;            /* is this a kermit window? */
X  int w_local;             /* is this a local window? */
X} WI_STR;
X
X#define LOG_BOTOM 1    /* print incoming characters */
X#define LOG_TOP       2    /* print lines which scroll off screen top */
X#define LOG_NONE  0    /* no printer output for this window */
X
Xtypedef struct gemfont {
X    short    ft_id;
X    short    ft_psize;
X    char ft_name[32];
X    short    ft_l_ade;
X    short    ft_h_ade;
X    short    ft_topl;
X    short    ft_ascl;
X    short    ft_halfl;
X    short    ft_descl;
X    short    ft_botl;
X    short    ft_chwidth;
X    short    ft_clwidth;
X    short    ft_lf_off;
X    short    ft_rt_off;
X    short    ft_thick;
X    short    ft_ul;
X    short    ft_light;
X    short    ft_skew;
X    short    ft_flags;
X    short    *ft_h_off;
X    short    *ft_c_off;
X    short    *ft_data;
X    short    ft_fwidth;
X    short    ft_fheight;
X    struct gemfont *ft_next;
X} GEMFONT;
X
Xtypedef char FUNCSTRING[MAXFUNCLEN];
X
SHAR_EOF
chmod 0600 WINDEFS.H || echo "restore of WINDEFS.H fails"
sed 's/^X//' << 'SHAR_EOF' > WINHASH.C &&
X/*
X * This file contains routines which can be used to determine the character
X * present at a given location in a window.  These functions are used to
X * determine the character under the mouse cursor, and for the log-top printer
X * function.
X */
X
X#include <stdio.h>
X#include <osbind.h>
```

```
X#include "windefs.h"
X
Xextern struct wi_str w[];
X
Xint highlighted_wdes = -1;  /* make this global now for use in winmain.c*/
X
X/*
X * char_at returns the ascii code of the character in wdes window at
X * (x_loc, y_loc) cursor position.  Zero is returned if the character can
X * not be identified.
X */
Xint char_at(wdes, x_loc, y_loc)
Xint wdes, x_loc, y_loc;
X
X{
X
X  register struct wi_str *wp = &w[wdes];
X  int ch, width, boffset, count, cur_y, cur_x;
X  register unsigned long mask;
X  register unsigned long *dptr;
X  register int shift;
X  register FNT *fnt = wp->font;
X  static char pa[32];      /* pixel array containing character image */
X
X  cur_x = x_loc * fnt->inc_x + X0;
X  cur_y = y_loc * fnt->inc_y + wp->top_y;
X
X  width = 2 * wp->wi_mf.wwords;
X  boffset = wp->m_off+cur_x+fnt->inc_x-1;
X  dptr = ((char *)(wp->wi_mf.ptr)) + cur_y * width
X    + ((boffset >> 4) << 1) - 2;
X  shift = 15 - (boffset & 15);
X  mask = (-1L<<(shift+fnt->inc_x)|(1<<shift)-1);
X  for (count = 0; count < fnt->inc_y; count ++)
X  {
X    pa[count] = (*dptr & (~ mask)) >> shift;
X    ((char *) dptr) += width;
X  }
X
X  ch = which_char(fnt, &fnt->f_hash, pa);
X  if (ch == 0)        /* Not found, check for inverse vidio case */
X  {
X    for (count = 0; count < fnt->inc_y; count ++)
X      pa[count] = ~pa[count] & (1<< fnt->inc_x) - 1;
X    ch = which_char(fnt, &fnt->f_hash, pa);
X  }
X  return (ch);
X}
X
X/*
X * which_char returns the ascii code of the character bitmap in pa from font
X * font using hash table htbl.
X */
```

```
Xint which_char(font, htbl, pa)
XFNT *font;
XHTBL *htbl;
Xchar *pa;
X
X{
X  register int i, h, n;
X
X  h = hash_char (pa, font->inc_y);
X  if ((htbl->h_t[h] & 128) == 0) /* collision bit set? */
X  {
X    for (i=0; i<font->inc_y; i++)
X      if (font->f_data[(htbl->h_t)[h] * 16 + i] != pa[i])
X        break;
X    if (i < font->inc_y){
X      return 0;
X    }
X    else
X      return (htbl->h_t[h]);
X  }
X  else                /* collision, search colision list for char */
X  {
X    n = htbl->h_t[h] & 127;
X    while (n != 0)
X    {
X      for (i=0; i<font->inc_y; i++)
X        if (font->f_data[(htbl->h_colision)[n].h_try * 16 + i] != pa[i])
X      break;
X      if (i < font->inc_y)
X        n = htbl->h_colision[n].h_next;
X      else
X        return (htbl->h_colision[n].h_try);
X    }
X    return 0;
X  }
X}
X
X/*
X * gen_hash initalizes htbl with the hash table for font font.
X */
Xvoid gen_hash(font, htbl)
XFNT *font;
XHTBL *htbl;
X
X{
X  register int overflowcnt = 1;
X  register int np, tp, i, j;
X  int h;
X
X  for (i=0; i<128; i++)         /* initalize */
X    htbl->h_colision[i].h_next = '\0';
X  for (i=0; i<(1<<HASHBITS); i++)
X    htbl->h_t[i] = '\0';
```

```
X
X  for (i=32; i<128; i++)
X  {          /* for each character */
X    h = hash_char(&font->f_data[i*16], font->inc_y);
X    if (htbl->h_t[h] == 0)
X      htbl->h_t[h] = i;
X    else
X    {
X      if ((htbl->h_t[h] & 128) == 0) /* already overflowed? */
X      { /* No, move first element to overflow list */
X    htbl->h_colision[overflowcnt].h_try = htbl->h_t[h];
X        htbl->h_t[h] = 128 | overflowcnt;
X    overflowcnt++;
X      }
X
X      tp = htbl->h_t[h] & 127;
X      j=0;   /* dbug */
X      while ((np = htbl->h_colision[tp].h_next) != 0){
X    tp = np;
X    j++; /* dbug */
X      }
X#ifdef DEBUG
X      printf("overflow=%d depth=%d hash %d = %d\n",overflowcnt, j, i, h);
X#endif
X      htbl->h_colision[tp].h_next = overflowcnt;
X      htbl->h_colision[overflowcnt].h_try = i;
X      ++ overflowcnt;
X    }
X  }
X
X#ifdef DEBUG      /* verify sanity of hash table */
X  for (i=32; i<128; i++) /* dbug */
X    if ((j = which_char(font, htbl, &font->f_data[i*16])) != i)
X  {
X    printf("%d != %d\n", i, j);
X  }
X#endif
X}
X
X/*
X * hash_char returns the hash code for the character pixel image in pa of
X * height height.
X */
Xint hash_char(pa, height)
Xchar *pa;
Xint height;
X
X{
X  register int h, i, j;
X
X  h=0;
X  for (i=0; i<height; i++)
X  {
```

```
X    h += pa[i] << i*3 % HASHBITS;
X  }
X  j = (1<<HASHBITS) - 1;
X  h = (h + (h >> HASHBITS + 1)) & j;
X
X  return h;
X}
X
X/*
X * dump_line send the y_coord'th line in the window wdes to the printer.
X * zero is returned if the line was printed ok.
X */
Xdump_line(wdes, y_coord)
X{
X  int i;
X  char ch;
X
X  for (i = 0; i < w[wdes].x_chrs; i++){
X    ch = char_at(wdes, i, y_coord);
X    if (Cprnout(ch) == 0) {
X      w[wdes].ptr_status = LOG_NONE;
X      return -1;
X    }
X  }
X  if (Cprnout('\n') == 0 | Cprnout('\r') == 0) {
X    w[wdes].ptr_status = LOG_NONE;
X    return -1;
X  }
X  return 0;
X}
X
X/*
X * dump_window send the contents of wdes to the printer.
X */
Xdump_window(wdes)
X{
X  int i;
X
X  for (i = 0; i < w[wdes].y_chrs; i++){
X    dump_line(wdes, i);
X  }
X}
X/* copy_word fills buffer with the word at (x1, y1) from wdes
X * No checks are made for buffer overflow.
X */
Xcopy_word(wdes, x1, y1, buffer)
Xint wdes, x1, y1;
Xchar * buffer;
X{
X  register int fx, lx, yo, xo;
X
X  yo = w[wdes].y_off / w[wdes].font->inc_y; /* adjust for scroll bar action */
X  xo = w[wdes].x_off / w[wdes].font->inc_x;
```

```
X  fx = x1 - 1;
X  lx = x1 + 1;
X
X  while (fx + xo >= 0 && char_at(wdes, fx + xo, y1 + yo) != ' ')
X    --fx;
X  while (lx + xo < w[wdes].x_chrs && char_at(wdes, lx + xo, y1 + yo) != ' ')
X    ++lx;
X  copy_text(wdes, ++fx, y1, --lx, y1, buffer);
X}
X
X/* copy_text fills buffer with the text between (x1, y1) and (x2, y2) from wdes
X * where point 1 is before point 2.  No checks are made for buffer overflow.
X */
Xcopy_text(wdes, x1, y1, x2, y2, buffer)
Xint wdes, x1, y1, x2, y2;
Xchar * buffer;
X{
X  static int old_hh;
X  static int old_y1;
X  register int cx, cy, i, lastnbi, lastnbx;
X  int inc_y, inc_x, top_y;
X
X  /* undo any old selected text which might be inverted on screen */
X  if (highlighted_wdes > 0 && w[highlighted_wdes].font != NULL)
X  {
X    w_update(highlighted_wdes, FM_COPY, 0, old_y1,
X      w[highlighted_wdes].x_chrs * w[highlighted_wdes].font->inc_x + X0 + 1,
X      old_hh);
X  }
X  if (y1 > y2) return; /* nothing to do */
X
X  inc_y = w[wdes].font->inc_y;
X  inc_x = w[wdes].font->inc_x;
X  top_y = w[wdes].top_y;
X
X  y1 = y1 + w[wdes].y_off / inc_y;  /* adjust for scroll bar action */
X  y2 = y2 + w[wdes].y_off / inc_y;  /* assumes _off is multiple of inc_ */
X  x1 = x1 + w[wdes].x_off / inc_x;
X  x2 = x2 + w[wdes].x_off / inc_x;
X  i = 0;
X  lastnbi = 0;
X  lastnbx = x1;
X
X  if (y1 > y2)
X  {
X    highlighted_wdes = -1; /* no undo required on next call */
X    return;
X  }
X
X  old_y1 = y1 * inc_y + top_y;
X  old_hh = (y2 - y1 + 2) * inc_y + Y0;
X  highlighted_wdes = wdes;
X
```

```
X  cx = x1;
X  cy = y1;
X
X  while (cy < y2 || (cy == y2 && cx <= x2))
X  {
X    buffer[i] = char_at(wdes, cx, cy);
X    if (buffer[i] != ' ')
X    {
X      lastnbi = i;
X      lastnbx = cx;
X    }
X    cx ++;
X    if (cx >= w[wdes].x_chrs)
X    {
X      if (cy == y1)    /* first line */
X        w_update(wdes, FM_INVERT, x1 * inc_x, cy * inc_y + top_y,
X      (++lastnbx - x1) * inc_x + X0, inc_y);
X      else
X        w_update(wdes, FM_INVERT, 0, cy * inc_y + top_y,
X      ++lastnbx * inc_x + X0, inc_y);
X      cx = 0;
X      cy ++;
X      i = ++lastnbi;
X      lastnbx = -1;
X      buffer[i] = '\r';
X    }
X    i++;
X  }
X  if (cy == y1)   /* one line */
X    w_update(wdes, FM_INVERT, x1 * inc_x, cy * inc_y + top_y,
X      (++lastnbx - x1) * inc_x + X0, inc_y);
X  else
X    w_update(wdes, FM_INVERT, 0, cy * inc_y + top_y,
X      ++lastnbx * inc_x + X0, inc_y);
X  buffer[++lastnbi] = '\0';
X}
X
SHAR_EOF
chmod 0600 WINHASH.C || echo "restore of WINHASH.C fails"
sed 's/^X//' << 'SHAR_EOF' > WINIO.C &&
X/*
X * This file contains file io routines
X */
X#include <obdefs.h>
X#include <gemdefs.h>
X#include <stdio.h>
X#include <osbind.h>
X#include <xbios.h>
X#include "wind.h"
X#include "uw.h"
X#include "windefs.h"
X
Xextern struct wi_str w[];
```

```
X
X/*
X * Character stuff (fonts, keymaps).
X */
Xextern FNT *fnttbl[10];             /* List of pointers to fonts avail */
Xextern int fontsavail;          /* Number of fonts available */
Xextern FNT *curfont;            /* font in use */
Xextern int fontmenuobj[];       /* array of font menu object numbers */
X/*
X * Options (somewhat loosely defined).
X */
Xextern int fast;           /* flag for fast open/close */
Xextern int overstrike;          /* flag for character output */
Xextern int sliders;             /* flag for sliders on new windows */
Xextern int titles;              /* flag for title bars on new windows */
Xextern int audibell;            /* Audible bell. */
Xextern int visibell;            /* Visible bell. */
Xextern int toponbel;            /* Top window on bell. */
X
Xextern FUNCSTRING fstrings[];       /* storage for function key bodys */
X
Xextern char cmdpath[];          /* default command executions strings */
Xextern char cmdname[];
Xextern char cmdargs[];
X
Xextern OBJECT *menubar;
X
Xvoid rsbufset();
Xlong    bufsizs[] = {      /* rs-232 buffer sizes setable from uw */
X    256l,    2048l,   16384l
X};
Xint bufsiz = 0;        /* index into table above of current size */
X
X#ifdef  LOCATE
Xextern   char *locate();
X#else
X#define  locate(n) n
X#endif
X
X/* read a font from a file */
XFNT * loadfont (name)
Xchar    *name;
X{
X  FNT * curfont;
X  char *tmpfnt;
X  char menustr[20];
X  char namestr[20];
X  register int shift;
X  register unsigned int mask;
X  register int    i;
X  char *getmem();
X
X  if ((curfont = (FNT *) getmem((long)sizeof(FNT))) == NULL)
```

```
X    return(NULL);
X  i = Fopen(locate(name), 0);
X  if (i<0)
X    return (NULL);
X  Fread(i, 2048L, curfont->f_data);
X  Fclose(i);
X  curfont->inc_x = MINMAX(curfont->f_data[16], 1, 8);
X  curfont->inc_y = MINMAX(curfont->f_data[32], 1, 16);
X  shift = 8-curfont->inc_x;
X  mask = (1<<curfont->inc_x)-1;
X  tmpfnt = curfont->f_data;
X  for (i=0; i<2048; i++)
X  {
X    *tmpfnt = (*tmpfnt>>shift)&mask;
X    tmpfnt++;
X  }
X  sscanf(name, "wind%3s", namestr);
X  sprintf(menustr, "%1.2d x %1.2d %s font", curfont->inc_x, curfont->inc_y,
X    namestr);
X  set_menu_string(menustr, fontmenuobj[fontsavail]);
X  gen_hash(curfont, &curfont->f_hash);
X
X  fnttbl[fontsavail] = curfont;
X  fontsavail++;
X  return (curfont);
X}
X
X/* This function reads the config file from name in directory path and sets
X * the apropriate variables.
X */
Xread_config(path, name)
Xchar *path;
Xchar *name;
X{
X  char file[100];
X  char *ptr;
X  char *status;
X  FILE *fd;
X  int i, j;
X  char *index(), *rindex();
X
X  strcpy(file, path);
X  ptr = rindex(file, '\\');
X  if (ptr != NULL)
X    *(ptr+1) = '\0';
X  strcat(file, name);
X
X  fd = fopen(file, "r");
X  if (fd == NULL)
X  {
X    if (*path != '\0')/* don't print error if in startup */
X      form_alert(1, "[2][Can't open config input file][ok]");
X    return;
```

```
X  }
X  status = fgets(file, 100,fd);
X  while (status != NULL)
X  {
X    char * endline;
X    if ((endline = rindex(file, '\n')) != NULL)
X      *endline = '\0';
X    if (!strncmp("audibell=", file, 9))
X      audibell = atoi(file + 9);
X    else if (!strncmp("visibell=", file, 9))
X      visibell = atoi(file + 9);
X    else if (!strncmp("topbell=", file, 8))
X      toponbel = atoi(file + 8);
X    else if (!strncmp("fast", file, 4))
X      ++fast;
X    else if (!strncmp("slow", file, 4))
X        fast = 0;
X    else if (!strncmp("overstrike", file, 4))
X      ++overstrike;
X    else if (!strncmp("nooverstrike", file, 4))
X      overstrike = 0;
X    else if (!strncmp("sliders", file, 7))
X      ++sliders;
X    else if (!strncmp("nosliders", file, 9))
X        sliders = 0;
X    else if (!strncmp("titles", file, 6))
X      ++titles;
X    else if (!strncmp("notitless", file, 8))
X        titles = 0;
X    else if (!strncmp("font=", file, 5))
X    {
X      i = atoi(file+5);
X      if (fontsavail > i)
X      {
X        for (j = 0; j < fontsavail; j++)
X          objc_change(menubar, fontmenuobj[j], 0, 0, 0, 0, 0, NONE, 0);
X        objc_change(menubar, fontmenuobj[i], 0, 0, 0, 0, 0, CHECKED, 0);
X        curfont = fnttbl[i];
X      }
X    }
X    else if (!strncmp("bufsiz=", file, 7))
X    {
X      i = atoi(file+7);
X      for (j=0; j < sizeof(bufsizs)/sizeof(bufsizs[0]); j++)
X        if (bufsizs[j] >= i)
X      break;
X      bufsiz = j;
X      rsbufset(j);
X    }
X    else if (file[0] == 'f' && file[1] >= '0' && file[1] <= '9'
X      && (file[2] == '=' || file[3] == '='))
X      {
X     i = atoi(file+1);
```

```
X    if (i < 1) i = 1;
X    if (i > NFSTRINGS - 2) i = NFSTRINGS - 2;
X    status = index(file, '=');
X    ptr = index(status, '\n');
X    if (ptr != NULL)
X      *ptr = '\0';
X    strcpy(fstrings[i-1], status+1);
X      }
X    else if (!strncmp("cmdpath=", file, 8))
X    strcpy(cmdpath, file+8);
X    else if (!strncmp("cmdname=", file, 8))
X    strcpy(cmdname, file+8);
X    else if (!strncmp("cmdargs=", file, 8))
X    strcpy(cmdargs, file+8);
X    status = fgets(file, 100, fd);
X  }
X  objc_change(menubar, VISIBELL, 0, 0, 0, 0, 0,
X    visibell? CHECKED: 0, 0);
X  objc_change(menubar, AUDIBELL, 0, 0, 0, 0, 0,
X    audibell? CHECKED: 0, 0);
X  objc_change(menubar, TOPONBEL, 0, 0, 0, 0, 0,
X    toponbel? CHECKED: 0, 0);
X  menu_icheck(menubar, MFAST, !fast);
X  menu_icheck(menubar, OVERSTRI, overstrike);
X  menu_icheck(menubar, WINSTYLE, sliders);
X  menu_icheck(menubar, WINTITLE, titles);
X}
X
X/* This function writes the config file name in directory path.
X */
Xwrite_config (path, name)
Xchar    *path;
Xchar    *name;
X{
X    char file[100];
X    char *ptr;
X    FILE *fd;
X    int  i;
X    char *rindex();
X
X    strcpy(file, path);
X    ptr = rindex(file, '\\');
X    if (ptr != NULL)
X         *(ptr + 1) = '\0';
X    strcat(file, name);
X
X    fd = fopen(file, "w");
X    if (fd == NULL)
X    {
X         form_alert(1, "[2][Can't open config output file][ok]");
X         return;
X    }
X
```

```
X    fprintf(fd, "audibell=%d\n", audibell);
X    fprintf(fd, "visibell=%d\n", visibell);
X    fprintf(fd, "topbell=%d\n", toponbel);
X    fprintf(fd, "%s\n", fast? "fast": "slow");
X    fprintf(fd, "%s\n", overstrike? "overstrike": "nooverstrike");
X    fprintf(fd, "%s\n", sliders? "sliders": "nosliders");
X    fprintf(fd, "%s\n", titles? "titles": "notitles");
X    fprintf(fd, "bufsiz=%ld\n", bufsizs[bufsiz]);
X    fprintf(fd, "cmdpath=%s\n", cmdpath);
X    fprintf(fd, "cmdname=%s\n", cmdname);
X    fprintf(fd, "cmdargs=%s\n", cmdargs);
X    for (i = 0; i < fontsavail; i++)
X         if (curfont == fnttbl[i])
X              fprintf(fd, "font=%d\n", i);
X
X    for (i = 0; i < NFSTRINGS - 2; ++i)
X         fprintf(fd, "f%d=%s\n", i + 1, fstrings[i]);
X    fclose(fd);
X}
X
Xsetcapture (wp)
XWI_STR  *wp;
X{
X    char full[80];
X    register char *cp1, *cp2;
X
X    if (wp->wi_lfd)
X    {
X         fclose(wp->wi_lfd);
X         wp->wi_lfd = NULL;
X    }
X    if (*wp->wi_fname)
X    {
X         cp1 = full;
X         for (cp2 = wp->wi_fpath; *cp2; *cp1++ = *cp2++);
X         while (--cp1 >= full && *cp1 != '\\');
X         strcpy(++cp1, wp->wi_fname);
X         if ((wp->wi_lfd = fopen(full, "a")) == NULL)
X         {
X              form_alert(1, "[2][Unable to open capture file.][ok]");
X              *wp->wi_fname = '\0';
X         }
X    }
X    w_rename(wp - w, NULL);
X}
X
X#define  DEFUCR   0x88     /* Async, 1 stop bit */
X
Xint speedobjs[] = {
X    BB1920,  BB960,   BB480,   BB360,   BB240,   BB200,
     BB180,   BB120,
X    BB60,    BB30,    BB20,    BB15,    BB13,    BB11,
     BB7, BB5
```

```
X};
Xint flowobjs[] = {
X    FCBNONE, FCBXON,       FCBRTS
X};
Xint parobjs[] = {
X    PBNONE,       PBODD,        PBEVEN
X};
Xint stopobjs[] = {
X    SBB1,         SBB15,        SBB2
X};
Xint bcbobjs[] = {
X    BCB8,         BCB7,         BCB6,         BCB5
X};
Xint bufobjs[] = {
X    BUFSIZ1, BUFSIZ2, BUFSIZ32
X};
X#ifdef  GETSPEED
X/* Baud rates 75 and 50 are not currently decoded. */
Xint timevals[] = {
X    1,   2,   4,   5,   8,   10,  11,
X    16,  32,  64,  96,  128, 143, 175
X};
X#endif
X
Xint speed = -1;        /* Don't know */
Xint flow = -1;
Xint ucr = -1;
Xstruct iorec old_iorec;
X
X/* load rs232 configuration into RSCONF dialog */
Xgetrsconf ()
X{
X    OBJECT   *obj;
X    register int  i;
X#ifdef  GETSPEED
X#define MFP  0xFFFFFA01L
X#define TDDR 36
X#define hz_200   ((unsigned long *) 0x4BAL)
X    int  tv;           /* Timer value */
X    int  maxtv;             /* Maximum timer value */
X    unsigned long endhz;        /* End of polling period */
X    long savessp;      /* Old stack pointer */
X#endif
X#ifdef  GETFLOW
X    struct   rsiorec  {
X         char fill0[32];
X         char rsmode;
X         char fill1[1];
X    };
X#endif
X
X#ifdef  GETSPEED
X    savessp = Super(0L);
```

```
X    maxtv = 0;
X    endhz = *hz_200 + 8;
X    while (*hz_200 < endhz)
X    {
X         tv = *((unsigned char *) (0xFFFFFA01L + 36));
X         if (tv > maxtv)
X              maxtv = tv;
X    }
X    (void) Super(savessp);
X    for (i = sizeof(timevals) / sizeof(timevals[0]);
X      --i >= 0 && timevals[i] != maxtv; );
X    if (i >= 0)
X         speed = i;
X#endif
X#ifdef  GETFLOW
X    flow = ((struct rsiorec *) Iorec(0))->rsmode;
X    if (flow == 3)              /* UW doesn't support both */
X         flow = 1;
X#endif
X#ifdef  GETUCR
X    /* Dev. Kit code suggests that Rsconf() returns old register vals */
X    ucr = (xbios(15, -1, -1, -1, -1, -1, -1) >> 24) & 0xFF;
X#endif
X    rsrc_gaddr(R_TREE, RSCONF, &obj);
X    for (i = sizeof(speedobjs) / sizeof(int); --i >= 0; )
X         obj[speedobjs[i]].ob_state = NORMAL;
X    for (i = sizeof(flowobjs) / sizeof(int); --i >= 0; )
X         obj[flowobjs[i]].ob_state = NORMAL;
X    for (i = sizeof(parobjs) / sizeof(int); --i >= 0; )
X         obj[parobjs[i]].ob_state = NORMAL;
X    for (i = sizeof(stopobjs) / sizeof(int); --i >= 0; )
X         obj[stopobjs[i]].ob_state = NORMAL;
X    for (i = sizeof(bcbobjs) / sizeof(int); --i >= 0; )
X         obj[bcbobjs[i]].ob_state = NORMAL;
X    for (i = sizeof(bufobjs) / sizeof(int); --i >= 0; )
X         obj[bufobjs[i]].ob_state = NORMAL;
X
X    if (speed >= 0)
X         obj[speedobjs[speed]].ob_state = SELECTED;
X
X    if (flow >= 0)
X         obj[flowobjs[flow]].ob_state = SELECTED;
X
X    if (ucr >= 0)
X    {
X         switch (ucr & 0x6) /* Parity */
X         {
X         default:
X              obj[PBNONE].ob_state = SELECTED;
X              break;
X         case 0x4:
X              obj[PBODD].ob_state = SELECTED;
X              break;
```

```
X         case 0x6:
X              obj[PBEVEN].ob_state = SELECTED;
X              break;
X         }
X
X         switch (ucr & 0x18)
X         {
X         default:
X              obj[SBB1].ob_state = SELECTED;
X              break;
X         case 0x10:
X              obj[SBB15].ob_state = SELECTED;
X              break;
X         case 0x18:
X              obj[SBB2].ob_state = SELECTED;
X              break;
X         }
X
X         obj[bcbobjs[(ucr >> 5) & 3]].ob_state = SELECTED;
X    }
X    obj[bufobjs[bufsiz]].ob_state = SELECTED;
X}
X
X/* set rs232 configuration from the RSCONF dialog */
Xsetrsconf ()
X{
X    register int  i;
X    OBJECT   *obj;
X    short    parity = 0;
X    short    stop = 0;
X    short    bcb = -1;
X
X    rsrc_gaddr(R_TREE, RSCONF, &obj);
X
X    for (i = sizeof(speedobjs) / sizeof(int); --i >= 0; )
X         if (obj[speedobjs[i]].ob_state & SELECTED)
X         {
X              speed = i;
X              break;
X         }
X
X    for (i = sizeof(flowobjs) / sizeof(int); --i >= 0; )
X         if (obj[flowobjs[i]].ob_state & SELECTED)
X         {
X              flow = i;
X              break;
X         }
X
X    for (i = sizeof(parobjs) / sizeof(int); --i >= 0; )
X         if (obj[parobjs[i]].ob_state & SELECTED)
X         {
X              parity = i + 1;
X              break;
```

```
X        }
X
X    for (i = sizeof(stopobjs) / sizeof(int); --i >= 0; )
X        if (obj[stopobjs[i]].ob_state & SELECTED)
X        {
X            stop = i + 1;
X            break;
X        }
X
X    for (i = sizeof(bcbobjs) / sizeof(int); --i >= 0; )
X        if (obj[bcbobjs[i]].ob_state & SELECTED)
X        {
X            bcb = i;
X            break;
X        }
X
X    for (i = sizeof(bufobjs) / sizeof(int); --i >= 0; )
X        if (obj[bufobjs[i]].ob_state & SELECTED)
X        {
X            bufsiz = i;
X            break;
X        }
X
X    if ((parity || stop || bcb >= 0) && ucr < 0)
X        ucr = DEFUCR;
X    if (parity)
X        ucr = (ucr & ~0x6) | (parity << 1);
X    if (stop)
X        ucr = (ucr & ~0x18) | (stop << 3);
X    if (bcb >= 0)
X        ucr = (ucr & ~0x60) | (bcb << 5);
X
X    Rsconf(speed, flow, ucr, -1, -1, -1);
X    rsbufset(bufsiz);
X}
X
Xvoid rsbufset(bufsiz)
Xint bufsiz;
X{
X    struct iorec *ioptr;
X    static char    *memalloced = NULL;
X    static int    oldbufsiz = 0;
X
X    ioptr = Iorec(0);
X    if (old_iorec.io_bufsiz == 0)
X        old_iorec = *ioptr;    /* save gem buffer */
X    if (oldbufsiz != bufsiz) {
X        char *mem;
X        oldbufsiz = bufsiz;
X        mem = (char *) Malloc(bufsizs[bufsiz]);
X        if (mem != NULL) {
X            if (memalloced != NULL)
X                Mfree(memalloced);
```

```
X             memalloced = mem;
X             ioptr->io_buff = mem;
X             ioptr->io_bufsiz = bufsizs[bufsiz];
X             ioptr->io_head = 0;
X             ioptr->io_tail = 0;
X             ioptr->io_low = bufsizs[bufsiz] / 4;
X             ioptr->io_high = ioptr->io_bufsiz - ioptr->io_low;
X         }
X    }
X}
SHAR_EOF
chmod 0600 WINIO.C || echo "restore of WINIO.C fails"
sed 's/^X//' << 'SHAR_EOF' > WINMAIN.C &&
X/* multi-window terminal emulation Version 0.2, May 29th 1986
X *
X * (C) Copyright 1986 Hans-Martin Mosner, University of Dortmund, Germany
X * You may freely use and distribute this program as source
X * and/or binary provided you make no profit with it.
X */
X
X#include <obdefs.h>
X#include <gemdefs.h>
X#include <osbind.h>
X#include <stdio.h>
X#include <bios.h>
X#include <xbios.h>
X#include "wind.h"
X#include "uw.h"
X#include "windefs.h"
X
X#define FIRSTOBJ DABOUT
X#define LASTOBJ PRTSTOP
X
X
X/* global constants
X */
X
Xextern  int      gl_apid;
Xextern  struct   wi_str   w[];
X
Xint contrl[12];
Xint intin[128];
Xint ptsin[128];
Xint intout[128];
Xint ptsout[128];
Xint work_in[11];
Xint work_out[57];
Xint pxyarray[20];
Xint msgbuff[8];
X
Xint phys_handle, handle;
X
X/*
```

```
X * Variables used by various routines (miscellaneous).
X */
X
Xlong    dummy;                  /* dummy return variable */
Xint scr_x, scr_y, scr_w, scr_h; /* size of screen */
Xint key;               /* the key pressed */
Xint tmp;               /* temporary for anything... */
Xchar    alert[300];            /* used for alerts */
Xint xsiz, ysiz;            /* temporarys for size of window */
X
X/*
X * Character stuff (fonts, keymaps).
X */
XFNT *fnttbl[10];           /* List of pointers to fonts avail */
Xint fontsavail;            /* Number of fonts available */
XFNT *tempfont;             /* font pointer */
Xchar    *oldshifted;           /* Pointer to old shifted key map. */
Xchar    *oldnormal;            /* Pointer to old normal key map. */
Xchar    *oldcapslock;          /* Pointer to old caps lock key map. */
Xint menu_key_map[0x80];         /* menu objects indexed by key map
X                          (alt key) read from object file */
Xint fontmenuobj[] =             /* array of font menu object numbers */
X  { FNTBIG,
X    FNTSYS,
X    FNTOWN,
X    FNTALT,
X    FNTTINY,
X    FNTUNKNW,
X    /* Add new font menu items here! */
X    0
X  };
Xextern  FNT  *loadfont();
X
X/*
X * State (loosely defined).
X */
Xint tick;                   /* used for flashing cursor */
Xint outport;           /* ports used with uw */
Xint inwind, outwind;       /* windows for input/output */
Xint uw_runs;           /* is uw running ? */
XFNT *curfont;          /* font in use */
Xint mouse;                 /* is mouse visible ? */
Xint menonoff;          /* Menu toggle. */
Xint m1inout;           /* mouse event  enter = 0  exit = 1 */
Xint sel_inp_mode;          /* input select with right button */
Xextern int highlighted_wdes;        /* window with text highligted */
X#define LOCKLEN 30             /* Max length of lock password */
Xchar    lockword[LOCKLEN];     /* lock password */
Xchar    lockbld[LOCKLEN];      /* lock password build area*/
Xextern int kermwdes;           /* port number for kermit window */
X
X/*
X * Options (somewhat loosely defined).
```

```
X */
Xint fast;                  /* flag for fast open/close */
Xint overstrike;            /* flag for character drawing */
Xint sliders;           /* flag for sliders on new windows */
Xint titles;                /* flag for titles on new windows */
Xint audibell;          /* Audible bell. */
Xint visibell;          /* Visible bell. */
Xint toponbel;          /* Top window on bell. */
X
XFUNCSTRING fstrings[NFSTRINGS];     /* storage for function key bodys */
X
Xchar    pastebuff[4096] = "";       /* 50 full lines of text */
X
Xchar    confpath[80] = ".\\*.*";    /* initial config file path */
Xchar    confname[40] = "win.cnf";   /* initial config file name */
Xint confbutt;          /* button from fsel_input */
X
X/*
X * Screen forms, objects, and other similar goodies...
X */
XMFDB    screen_mf;
X
XOBJECT  *obj_tmp, *menubar;
XTEDINFO *ted_tmp;
XTEDINFO *ted_fnum;
X
Xextern struct iorec old_iorec;  /* copy of old rs_232 io record from winio.c */
X
XMFORM   rmbmform[1];
XMFORM   lckmform[1];
X
X#define  CHECKWIN  if (!outwind) break
X
X/*
X * The program code...
X */
X
Xextern   char *getmem();
X#ifdef   MWC
X#include <linea.h>
X#endif
X
X/*
X * If LOCATE is defined, it is assumed to be the path to check after
X * "." for standard files (fonts, config files, etc.).
X */
X#ifndef  LOCATE
X#define  locate(n) n
X#else
Xchar    locpath[64] = LOCATE;
X
Xchar    *
Xlocate (name)
```

```
Xchar    *name;
X{
X    int  fd;
X    static   char new[80];
X
X    close(fd = open(name, 0));
X    if (fd >= 0)
X         return (name);
X
X    sprintf(new, "%s\\%s", locpath, name);
X    close(fd = open(new, 0));
X    return (fd >= 0? new: name);
X}
X#endif
X
Xstartup()
X{
Xint i,j;
Xchar *tmpfnt;
Xstatic char my_shft_map[128];
Xstatic char my_norm_map[128];
X
X#ifndef  MWC
Xregister GEMFONT **a5;     /* this is really register A5 */
X#endif
X
X  /* start up everything: appl_init, menu, and the like
X   */
X  appl_init();
X  rsrc_load(locate("wind.rsc"));
X  rsrc_gaddr(R_TREE, MENUBAR, &menubar);
X
X  /*
X   * Fill menu_key_map by scaning object strings for ALTINDICATOR.
X   */
X  for (i = FIRSTOBJ; i <= LASTOBJ; i++)
X    if (menubar[i].ob_type == G_STRING)
X    {
X      char * found;
X      char * index();
X
X      found = index((char *)menubar[i].ob_spec, ALTINDICATOR);
X      if (found != NULL)
X      {
X        menu_key_map[*(++found) & 0x7f] = i;
X#ifdef DEBUG
X        printf("menu_key_map[%c] = %d\n", *found, i);
X#endif
X      }
X    }
X
X/*  objc_change(menubar, VISIBELL, 0, 0, 0, 0, 0, CHECKED, 0); *already done*/
X  menu_bar(menubar, menonoff = 1);
```

```
X  m1inout = 1;
X  audibell = 1;
X  visibell = 1;
X  sliders = 1;
X  titles = 1;
X
X  /* set mouse symbol to arrow
X   */
X  graf_mouse(ARROW, NULL);
X  mouse = 1;
X
X  /* get screen handle and sizes;
X   * open virtual workstation
X   */
X  phys_handle = graf_handle(&dummy, &dummy, &dummy, &dummy);
X  wind_get(0, WF_WORKXYWH, &scr_x, &scr_y, &scr_w, &scr_h);
X  for (i=0; i<10; work_in[i++]=1);
X  work_in[10] = 2;
X  handle = phys_handle;
X  v_opnvwk(work_in, &handle, work_out);
X
X  /* set up screen mfdb
X   */
X
X  screen_mf.ptr = NULL;
X  screen_mf.wpix = 640;
X  screen_mf.hpix = 400;
X  screen_mf.wwords = 40;
X  screen_mf.format = 0;
X  screen_mf.planes = 1;
X
X  /*
X   * now set up the system fonts
X   */
X  fontsavail = 0;
X  if ((curfont = getmem((long)sizeof(FNT))) != NULL)
X  {
X#ifdef  MWC
X    linea0();
X    tmpfnt = la_init.li_a1[2]->font_data;     /* 8 x 16 system font */
X#else
X    asm("dc.w $a000");     /* get font addresses from line A */
X    asm("move.l a1,a5");        /* put it into A5 */
X    tmpfnt = a5[2]->ft_data;
X#endif
X    for (i=0; i<128; i++)
X      for (j=0; j<16; j++)
X        curfont->f_data[i*16+j] = tmpfnt[i+j*256];
X    curfont->inc_x = 8;
X    curfont->inc_y = 16;
X
X    set_menu_string("8 x 16 sys font", fontmenuobj[fontsavail]);
X
```

```
X    curfont->def_win_x = 80;    /* Set default window size */
X    curfont->def_win_y = 24;
X
X    gen_hash(curfont, &curfont->f_hash);
X
X    fnttbl[fontsavail] = curfont;
X    fontsavail++;
X  }
X  /* set up 6 x 6 system font */
X  if ((curfont = getmem((long)sizeof(FNT))) != NULL)
X  {
X#ifdef  MWC
X    linea0();
X    tmpfnt = la_init.li_a1[0]->font_data;
X#else
X    asm("dc.w $a000");     /* get font addresses from line A */
X    asm("move.l a1,a5");       /* put it into A5 */
X    tmpfnt = a5[0]->ft_data;
X#endif
X    for (i=0; i<128; i++){ /* for each character (first 128) */
X      int bitpos, index, offset;
X
X      bitpos = i*6;
X      index = bitpos >> 3;
X      offset = bitpos & ~(~0<<3);
X      for (j=0; j<6; j++) /* for each pixel row in character */
X        curfont->f_data[i*16+j+1] = ((tmpfnt[index+j*192] << offset) +
X      (tmpfnt[index+j*192+1] >> (8-offset) & ~(~0<<offset)))>>2 & 63;
X      curfont->f_data[i*16+0] = 0; /* extra empty row */
X    }
X    curfont->inc_x = 6;
X    curfont->inc_y = 7;
X
X    /*
X     * The 6x6 system font can be made better.  (dot the i etc.)
X     * This is a bad hack, but it makes it easier to read.
X     */
X    curfont->f_data[105*16+1] = 8;
X    curfont->f_data[105*16+2] = 0;
X    curfont->f_data[105*16+5] = 8;
X    curfont->f_data[106*16+5] = 20;
X    curfont->f_data[106*16+6] = 8;
X    curfont->f_data[106*16+2] = 0;
X
X    curfont->def_win_x = 80;    /* Set default window size */
X    curfont->def_win_y = 40;
X
X    set_menu_string("6 x 7 sys font", fontmenuobj[fontsavail]);
X    gen_hash(curfont, &curfont->f_hash);
X
X    fnttbl[fontsavail] = curfont;
X    fontsavail++;
X  }
```

```
X
X  /*
X   * Load font file(s) if any.
X   */
X
X  if (curfont = loadfont("windstd.fnt")) /* yes, I want the assignment (=). */
X  {
X    objc_change(menubar, fontmenuobj[fontsavail-1], 0, 0, 0, 0, 0, CHECKED, 0);
X    curfont->def_win_x = 80;    /* Set default window size */
X    curfont->def_win_y = 24;
X  }
X  else {
X    objc_change(menubar, FNTSYS, 0, 0, 0, 0, 0, CHECKED, 0);
X    curfont = fnttbl[fontsavail - 1];
X  }
X
X  if (tempfont = loadfont("windalt.fnt"))
X  {
X    tempfont->def_win_x = 80;   /* Set default window size */
X    tempfont->def_win_y = 32;
X  }
X
X  if (tempfont = loadfont("windtny.fnt"))
X  {
X    tempfont->def_win_x = 80;   /* Set default window size */
X    tempfont->def_win_y = 48;
X  }
X  if (tempfont = loadfont("windoth.fnt"))
X  {
X    tempfont->def_win_x = 80;   /* Set default window size */
X    tempfont->def_win_y = 24;
X  }
X
X  /*
X   * Disable unused font entrys.
X   */
X  for (i = fontsavail; fontmenuobj[i] != 0; i++)
X  {
X    set_menu_string("unavailable font", fontmenuobj[i]);
X    objc_change(menubar, fontmenuobj[i], 0, 0, 0, 0, 0, DISABLED, 0);
X  }
X
X
X  /* set up the serial line (XON/XOFF flow control)
X   */
X
X  Rsconf(-1, 1, -1, -1, -1, -1);
X
X  /* Initalize function keys to default values.
X   */
X  for (i=0; i<NFSTRINGS; i++)
X  {
X    sprintf(fstrings[i], "^A%d^M", i);
```

```
X  }
X  fstrings[NFSTRINGS-1][0] = '\0'; /* last function zero length for coding */
X
X  /* Set up the key tables to use the help key for break.
X   */
X  oldshifted = ((struct keytbl *) Keytbl(-1L, -1L, -1L))->kt_shifted;
X  oldnormal = ((struct keytbl *) Keytbl(-1L, -1L, -1L))->kt_normal;
X  oldcapslock = ((struct keytbl *) Keytbl(-1L, -1L, -1L))->kt_capslock;
X
X  for (i=0; i<128; i++) { /* copy bios tables */
X    my_shft_map[i] = oldshifted[i];
X    my_norm_map[i] = oldnormal[i];
X  }
X
X  my_shft_map[KC_HELP] = 0x6f; /* except help key */
X  for (i=KC_F1; i<=KC_F10; i++) { /* and function keys */
X    my_norm_map[i] = 0x6f;
X  }
X#ifdef COMMENT
X  /*
X   * These keys must be decoded manualy so that the field editing
X   * of gem will not be impacted.
X   */
X  my_norm_map[KC_CDOWN] = 0x0a; /* and cursor motion keys (for adm31) */
X  my_norm_map[KC_CLEFT] = 0x08;
X  my_norm_map[KC_CRIGHT] = 0x0c;
X  my_norm_map[KC_CUP] = 0x0b;
X#endif
X/* (void) Keytbl(my_norm_map, my_shft_map, my_norm_map); */
X  (void) Keytbl(my_norm_map, my_shft_map, -1L);
X
X  form_dial(FMD_START, 0, 0, 0, 0, scr_x, scr_y, scr_w, scr_h);
X  form_dial(FMD_FINISH, 0, 0, 0, 0, scr_x, scr_y, scr_w, scr_h);
X
X  /*
X   * Read initial configuration file if any
X   */
X  read_config("", locate(confname));
X}
X
Xfinish()
X{
Xint i;
Xstruct iorec *ioptr;
X
X  /* close all windows and clean up
X   */
X
X  if (uw_runs) xmitcmd(CB_FN_MAINT|CB_MF_EXIT);
X  for (i=0; i<MAX_WIND; i++)
X    w_close(i);
X  (void) Keytbl(oldnormal, oldshifted, oldcapslock); /* restore key maps */
X
```

```
X  if (old_iorec.io_bufsiz != 0) {
X    ioptr = Iorec(0); /* restore rs_232 io record */
X    *ioptr = old_iorec;
X  }
X
X  menu_bar(menubar, 0);
X  v_clsvwk(handle);
X  appl_exit();
X  exit();
X}
X
Xmemory()
X{
Xlong m = Malloc(-1L);
X
X  sprintf(alert, "[0][Free memory: %ld bytes][Ok]", m);
X  form_alert(1, alert);
X}
X
Xmain()
X{
X  long _stksiz = 4096l;         /* Tell Mark Williams C we need 4k stack */
X  int   event;
X  int   menuitem = 0;      /* Alt sequence mapped to menu item */
X  int   cx, cy, cw, ch;
X  int   mx, my, mb, mk;         /* mouse coordinates from event_multi */
X  int   ww, wh;            /* window width & height for move */
X  int   buttonstate = 2;   /* button state to wait for next. */
X  int   clicks;            /* number of clicks seen by evnt_multi. */
X  register int    cnt;
X  int   funcindex = 0;
X  int   regionflag = 0;         /* indicates region selection in progress */
X  int   locked = 0;        /* keyboard and mouse locked with password ?*/
X  startup();
X
X  xsiz = curfont->def_win_x;
X  ysiz = curfont->def_win_y;
X
Xinit:
X  w_open(1, "Terminal", xsiz, ysiz);
X  outport = 1;
X  inwind = outwind = find_wind(1);
X  printer_mark(outwind);
X
X  for (;;)
X  {
X    if (!mouse)
X    {
X      graf_mouse(M_ON, NULL);
X      ++mouse;
X    }
X    if (menuitem)
X    {
```

```
X      event = MU_MESAG;
X      msgbuff[0] = MN_SELECTED;
X      msgbuff[4] = menuitem;
X      menuitem = 0;
X    }
X    else
X    {
X      event = evnt_multi(MU_MESAG | MU_KEYBD | MU_TIMER | MU_BUTTON | MU_M1,
X    2, 2, buttonstate,
X    m1inout, scr_x, scr_y, scr_w, scr_h,
X    0, 0, 0, 0, 0,
X    msgbuff, 15, 0,
X    &mx, &my, &mb, &mk,
X    &key, &clicks);
X    }
X
X    if (event & MU_KEYBD)
X    if (locked) {
X      static char lockkey[LOCKLEN];
X      static int lockindex;
X      int keyval = key & 0xff;
X
X      if (keyval == '\r') {
X        lockkey[lockindex] = '\0';
X        if (strcmp (lockkey, lockword) == 0) {
X          graf_mouse(ARROW, &dummy);
X          menu_bar(menubar, menonoff);
X      locked = 0;
X    }
X    lockindex = 0;
X      }
X      else
X        if (lockindex < LOCKLEN - 1)
X          lockkey[lockindex++] = keyval;
X    }
X    else
X    {
X      int i, j;
X      int outcount;
X      int keyindex = key >> 8;
X      int keyval = key & 0xff;
X      int funcindex = NFSTRINGS;
X      char outputstring[MAXFUNCLEN];
X
X#ifdef DEBUG
X      printf("keyindex = %x, keyval = %x\n", keyindex, keyval);
X#endif
X
X      /*
X       * Decode key for function keys or other special keys
X       * This section could be made faster at the expense of less clear
X       * code.  It seems to be fast enough for now. (How fast can you type?)
X       * It should probrably be table driven.
```

```
X       */
X      if (keyindex >= KC_F1 || (keyval == 0
X        && keyindex != 0x39 /* ctl-space */ && keyindex != 3 /* ctl-@ */))
X      { /* decode these keys manualy */
X        if (keyindex >= KC_F1 && keyindex <= KC_F10) /* Unshifted F keys */
X        {
X      if (keyval == 0x6f)  /* plain function key */
X        funcindex = keyindex - KC_F1;
X
X      else if (keyval == 0x0f)/* control function key */
X        funcindex = keyindex - KC_F1 + 10;
X
X      else if (keyval == 0) /* alt function key */
X        funcindex = keyindex - KC_F1 + 30;
X        }
X
X        else if (keyindex >= KC_F1+25 && keyindex <= KC_F10+25) /* shift F */
X          funcindex = keyindex - KC_F1 - 25 + 20;
X
X    else if (key == (KC_CDOWN << 8))
X      keyval = 0x0a;
X
X    else if (key == (KC_CLEFT << 8))
X      keyval = 0x08;
X
X    else if (key == (KC_CRIGHT << 8))
X      keyval = 0x0c;
X
X    else if (key == (KC_CUP << 8))
X      keyval = 0x0b;
X
X        else if (key == 0x626f) /* break key (shift help) */
X        {
X          long    clk_time;
X
X      clk_time = clock();
X      Rsconf(-1, -1, -1, -1, 9, -1); /* set break condition */
X      while (clock() - clk_time < BREAK_TICKS); /* delay while break is sent */
X      Rsconf(-1, -1, -1, -1, 1, -1); /* clear break condition */
X      funcindex = NFSTRINGS - 1; /* last function key is zero length */
X    }
X    else if (keyval == 0)
X    {    /* check if alt key used to select menu entry */
X      if (keyindex >= 0x78 && keyindex <= 0x7e)
X      { /* alt number used to select window number */
X        int newwind = keyindex - 0x77;
X
X        if (find_port(newwind) > 0)
X          w_top(newwind);
X      }
X      else
X      { /* alt key short-hand--fake menu message next time around */
X        menuitem = menu_key_map[oldnormal[keyindex] & 0x7f];
```

```
X#ifdef DEBUG
X        printf("menu keyindex = %x, char = %c, menuitem = %d\n",
X          keyindex, oldnormal[keyindex], menuitem);
X#endif
X      }
X      funcindex = NFSTRINGS - 1; /* last function key is zero length */
X    }
X      }
X      if (funcindex < NFSTRINGS) /* if function key */
X      {      /* copy function string to outputstring decoding ^ */
X    j=0;
X    outcount = 0;
X        for (i=0; fstrings[funcindex][i] != '\0'; i++)
X    {
X      outputstring[j] = fstrings[funcindex][i];
X      if (outputstring[j] == '^')
X      {
X        i++;
X        if (fstrings[funcindex][i] == '?')
X          outputstring[j] = '\177';
X        else if (fstrings[funcindex][i] != '^')
X          outputstring[j] = fstrings[funcindex][i] & 0x1f;
X      }
X      j++;
X    }
X    outcount = j;
X      }
X      else   /* normal key press */
X      {
X        outputstring[0] = keyval & 0x7f;
X    outcount = 1;
X      }
X      /*
X       * Now output outcount characters which are stored in outputstring.
X       */
X      proto_out(outport, outputstring, outcount);
X    }
X
X    if (event & MU_TIMER)
X    {
X      if (tick++ > 20)
X      {
X    w_flash(outwind, 2);
X    tick = 0;
X    if (kermwdes) kermtimchk();
X      }
X      /* act on any input from serial port */
X      if (proto_in () < 0) goto init/* uw mode ended? */;
X    }
X
X    if (event & MU_MESAG)
X    {
X      switch (msgbuff[0])
```

```
X      {
X      case MN_SELECTED:
X    switch (msgbuff[4])
X    {
X    case REMSHELL:
X    case SHELLOTH:
X    case SHELL32:
X    case SHELL24:
X    case SHELL12:
X    case SHELL8:
X      if (msgbuff[4] == REMSHELL)
X      {  /* default window size */
X            xsiz = curfont->def_win_x;
X        ysiz = curfont->def_win_y;
X      }
X      else if (msgbuff[4] == SHELLOTH)
X      {
X        if (!size_dial()) break;
X      }
X      else
X      {  /* size specified in menu string */
X        char *ent_str;
X
X        ent_str = (char *) menubar[msgbuff[4]].ob_spec;
X        sscanf(ent_str, "%*s%d%*s%d", &ysiz, &xsiz);
X      }
X      if (uw_runs)
X      {
X        tmp = w_open(0, "Terminal", xsiz, ysiz);
X        if (tmp)
X        {
X          outport = tmp;
X          outwind = find_wind(tmp);
X          xmitcmd(CB_FN_NEWW|tmp);
X        }
X      } else
X      {
X        form_alert(1, "[1][UW is not running][Ok]");
X      }
X      break;
X
X    case LOCCOPY:
X      for (cnt=1; cnt<MAX_WIND; cnt++)
X      {
X        if (!find_wind(cnt))
X        {
X          break;       /* a free port */
X        }
X      }
X      if (cnt < MAX_WIND)
X      {
X        static int c[8];
X        struct wi_str *wp1 = &w[outwind];
```

```
X       struct wi_str *wp2;
X       outport = w_open(cnt, "Local Copy", wp1->x_chrs, wp1->y_chrs);
X       outwind = find_wind(outport);
X       wp2 = &w[outwind];
X       c[0] = wp1->m_off;
X       c[1] = wp1->top_y;
X       c[2] = c[0] + wp1->wi_mf.wpix;
X       c[3] = c[1] + wp1->wi_mf.hpix;
X       c[4] = wp2->m_off;
X       c[5] = wp2->top_y;
X       c[6] = c[4] + wp2->wi_mf.wpix;
X       c[7] = c[5] + wp2->wi_mf.hpix;
X       vro_cpyfm(handle, FM_COPY, c, &wp1->wi_mf, &wp2->wi_mf);
X       wp2->w_local = TRUE;
X     }
X     break;
X
X   case WINRESIZ:
X     if (!size_dial()) break;
X     w_resize(outwind, xsiz, ysiz);
X     break;
X
X   case FQUIT:
X     finish();
X
X   case FQUITUW:
X     if (uw_runs)
X     {
X     int i;
X       xmitcmd(CB_FN_MAINT|CB_MF_EXIT);
X       uw_runs = 0;
X       for (i=1; i<8; i++)
X       {
X         w_close(find_wind(i));
X       }
X       menu_tnormal(menubar, msgbuff[3], 1);
X       goto init;
X     }
X     break;
X
X   case LOADCONF:
X     fsel_input(confpath, confname, &confbutt);
X     if (confbutt)
X       read_config(confpath, confname);
X     break;
X
X   case SAVECONF:
X     fsel_input(confpath, confname, &confbutt);
X     if (confbutt)
X       write_config(confpath, confname);
X     break;
X
X   case CAPTURE:
```

```
X      if (outwind)
X      {
X        WI_STR *wp = &w[outwind];
X
X        fsel_input(wp->wi_fpath, wp->wi_fname, &confbutt);
X        if (confbutt)
X          setcapture(wp);
X      }
X      break;
X
X    case COMMAND:
X      do_exec();
X      break;
X
X    case SETPATH:
X      do_path();
X      break;
X
X    case KERMIT:
X      kerminit(outwind);
X      break;
X
X    case FLOCKKEY:
X      s_dial(LOCKINFO, 1);
X      cnt = 0;
X      while ((key = evnt_keybd() & 0xff) != '\r') {
X        if (key == '\033') {    /* ESC means use old password */
X          locked = 1;
X          strcpy(lockbld, lockword);
X          break;
X        }
X        if (cnt < LOCKLEN - 1)
X          lockbld[cnt++] = key;
X      }
X      while (! locked) {
X        lockbld[cnt] = '\0';
X        locked = 1;
X        cnt = 0;
X        while ((key = (evnt_keybd() & 0xff)) != '\r')
X          if (cnt < LOCKLEN - 1)
X            if (key != lockbld[cnt++]) {
X           locked = 0;
X           lockbld[cnt - 1] = key;
X         }
X        if (lockbld[cnt] != '\0')
X          locked = 0;
X      }
X      if (lockbld[0] == '\0')
X        locked = 0;
X      if (locked) {
X        strcpy(lockword, lockbld);
X        menu_bar(menubar, 0);
X        graf_mouse(USER_DEF, lckmform);
```

```
X      }
X      s_dial(LOCKINFO, 2);
X      break;
X
X    case DABOUT:
X      if (s_dial(ABOUT, 3) == MOREINF1)
X      {
X        if (s_dial(INFO1, 3) == CONT1)
X        {    /* give short help screens */
X          if (s_dial(INFO2, 3) == CONT2)
X          {
X            if (s_dial(INFO3, 3) == CONT3)
X            {
X              if (s_dial(INFO4, 3) == CONT4)
X              {
X             s_dial(INFO5, 3);
X              }
X            }
X          }
X        }
X      }
X      break;
X
X    case WRENAME:
X      CHECKWIN;
X      rsrc_gaddr(R_TREE, NEWNAME, &obj_tmp);
X      ted_tmp = (TEDINFO *) obj_tmp[FLD1].ob_spec;
X      strcpy(ted_tmp->te_ptext, w[outwind].name);
X      form_center(obj_tmp, &cx, &cy, &cw, &ch);
X      form_dial(FMD_START, 0, 0, 20, 10, cx, cy, cw, ch);
X      if (!fast) form_dial(FMD_GROW, 0, 0, 20, 10, cx, cy, cw, ch);
X      objc_draw(obj_tmp, 0, 5, cx, cy, cw, ch);
X      tmp = form_do(obj_tmp, FLD1);
X      if (!fast) form_dial(FMD_SHRINK, 0, 0, 20, 10, cx, cy, cw, ch);
X      form_dial(FMD_FINISH, 0, 0, 20, 10, cx, cy, cw, ch);
X      objc_change(obj_tmp, tmp, 0, cx, cy, cw, ch, NONE, 0);
X      if (tmp == OKRENAME) w_rename(outwind, (char *) ted_tmp->te_ptext);
X      break;
X
X    case FUNCTKEY:
X      rsrc_gaddr(R_TREE, FUNCTEDT, &obj_tmp);
X      ted_fnum = (TEDINFO *) obj_tmp[FUNCNAME].ob_spec;
X      sprintf(ted_fnum->te_ptext, "%2d", funcindex+1);
X      ted_tmp = (TEDINFO *) obj_tmp[FUNCBODY].ob_spec;
X      strcpy((char *) ted_tmp->te_ptext, fstrings[funcindex]);
SHAR_EOF
echo "End of part 3, continue with part 4"
echo "4" > s2_seq_.tmp
exit 0


From -845995821485941634
X-Google-Language: ENGLISH,ASCII-7-bit
```

X-Google-Thread: 100493,5f732513c2832d0f,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-02 14:06:57 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!csd4.milw.wisc.edu!bionet!agate!
saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i022: uw -- Unix Windows windowing terminal software part04/05
Keywords: shar
Message-ID: <6554@saturn.ucsc.edu>
Date: 2 Mar 89 22:06:57 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1827
Approved: koreth@ssyx.ucsc.edu

Submitted-by: sun.com!laidbak!katzung (Brian Katzung)
Posting-number: Volume 2, Issue 22
Archive-name: uw/part04

```
#!/bin/sh
# this is part 4 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file WINMAIN.C continued
#
CurArch=4
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> WINMAIN.C
X      form_center(obj_tmp, &cx, &cy, &cw, &ch);
X      form_dial(FMD_START, 0, 0, 20, 10, cx, cy, cw, ch);
X      if (!fast) form_dial(FMD_GROW, 0, 0, 20, 10, cx, cy, cw, ch);
X      objc_draw(obj_tmp, 0, 5, cx, cy, cw, ch);
X      tmp = 0;
X      while (tmp != FUNCEXIT)
X      {
X        int i;
X
X        tmp = form_do(obj_tmp, FUNCNAME);
X        switch (tmp)
X        {
X        case FUNCSHOW:
X          funcindex = atoi(ted_fnum->te_ptext) - 1;
X          if (funcindex < 0)
X            funcindex = NFSTRINGS - 3;
X          else if (funcindex > NFSTRINGS - 3)
X            funcindex = 0;
```

```
X         strcpy((char *) ted_tmp->te_ptext, fstrings[funcindex]);
X         objc_draw(obj_tmp, FUNCBODY, 5, cx, cy, cw, ch);
X         sprintf(ted_fnum->te_ptext, "%2d", funcindex+1);
X         objc_draw(obj_tmp, FUNCNAME, 5, cx, cy, cw, ch);
X         break;
X       case FUNCPREV:
X         funcindex --;
X         if (funcindex < 0)
X           funcindex = NFSTRINGS - 3;
X         sprintf(ted_fnum->te_ptext, "%2d", funcindex+1);
X         objc_draw(obj_tmp, FUNCNAME, 5, cx, cy, cw, ch);
X         strcpy((char *) ted_tmp->te_ptext, fstrings[funcindex]);
X         objc_draw(obj_tmp, FUNCBODY, 5, cx, cy, cw, ch);
X         break;
X       case FUNCNEXT:
X         funcindex ++;
X         if (funcindex > NFSTRINGS - 3)
X           funcindex = 0;
X         sprintf(ted_fnum->te_ptext, "%2d", funcindex+1);
X         objc_draw(obj_tmp, FUNCNAME, 5, cx, cy, cw, ch);
X         strcpy((char *) ted_tmp->te_ptext, fstrings[funcindex]);
X         objc_draw(obj_tmp, FUNCBODY, 5, cx, cy, cw, ch);
X         break;
X       case FUNCENT:
X         funcindex = atoi(ted_fnum->te_ptext) - 1;
X         strcpy(fstrings[funcindex], ted_tmp->te_ptext);
X         fstrings[funcindex][i] = ted_tmp->te_ptext[i];
X         strcpy((char *) ted_tmp->te_ptext, fstrings[funcindex]);
X         objc_draw(obj_tmp, FUNCBODY, 5, cx, cy, cw, ch);
X         break;
X       }
X       objc_change(obj_tmp, tmp, 0, cx, cy, cw, ch, NONE, 1);
X     }
X     if (!fast) form_dial(FMD_SHRINK, 0, 0, 20, 10, cx, cy, cw, ch);
X     form_dial(FMD_FINISH, 0, 0, 20, 10, cx, cy, cw, ch);
X     objc_change(obj_tmp, tmp, 0, cx, cy, cw, ch, NONE, 0);
X     break;
X
X   case MFREE:
X     memory();
X     break;
X
X   case MFAST:
X     menu_icheck(menubar, MFAST, fast);
X     fast = !fast;
X     break;
X
X   case OVERSTRI:
X     overstrike = !overstrike;
X     menu_icheck(menubar, OVERSTRI, overstrike);
X     break;
X
X   case FNTSYS:
```

```
X    case FNTOWN:
X    case FNTALT:
X    case FNTBIG:
X    case FNTTINY:
X    case FNTUNKNW:
X    /* Add new font menu items here! */
X      for (cnt = 0; fontmenuobj[cnt] != 0; cnt++)
X      {
X        if (fontmenuobj[cnt] != msgbuff[4])
X        {
X          if (cnt < fontsavail)
X            objc_change(menubar, fontmenuobj[cnt], 0, 0, 0, 0, 0,
X           NONE, 0);
X          else
X            objc_change(menubar, fontmenuobj[cnt], 0, 0, 0, 0, 0,
X           DISABLED, 0);
X        }
X        else
X          curfont = fnttbl[cnt];
X      }
X      objc_change(menubar, msgbuff[4], 0, 0, 0, 0, 0, CHECKED, 0);
X      break;
X
X    case PRTBOTOM:
X      if (outwind)
X      {
X        w[outwind].ptr_status = LOG_BOTOM;
X        printer_mark(outwind);
X        w_rename(outwind, NULL);
X      }
X      break;
X
X    case PRTTOP:
X      if (outwind)
X      {
X        w[outwind].ptr_status = LOG_TOP;
X        printer_mark(outwind);
X        w_rename(outwind, NULL);
X      }
X      break;
X
X    case PRTWIND:
X      if (outwind)
X      {
X        printer_mark(outwind);
X        dump_window(outwind);
X      }
X      break;
X
X    case PRTSTOP:
X      if (outwind)
X      {
X        w[outwind].ptr_status = LOG_NONE;
```

```
X       printer_mark(outwind);
X       w_rename(outwind, NULL);
X     }
X     break;
X
X   case INPUTWIN:
X   case PASTE:
X     sel_inp_mode = msgbuff[4];
X     graf_mouse(USER_DEF, rmbmform);
X     break;
X
X   case RESETAUX:
X     if (rs232_reset())
X       Cauxout(0x11); /* send XOFF  */
X     break;
X
X   case ASSRD:        /* Assert RTS/DTR. */
X        Offgibit(~0x18);
X        break;
X
X   case RESRD:        /* Reset RTS/DTR. */
X        Ongibit(0x18);
X        break;
X
X   case AUDIBELL:
X     audibell = !audibell;
X     objc_change(menubar, AUDIBELL, 0, 0, 0, 0, 0,
X       audibell? CHECKED: 0, 0);
X     break;
X
X   case VISIBELL:
X     visibell = !visibell;
X     objc_change(menubar, VISIBELL, 0, 0, 0, 0, 0,
X       visibell? CHECKED: 0, 0);
X     break;
X
X   case TOPONBEL:
X     toponbel = !toponbel;
X     objc_change(menubar, TOPONBEL, 0, 0, 0, 0, 0,
X       toponbel? CHECKED: 0, 0);
X     break;
X
X   case SHRNKWIN:     /* Shrink window to iconic size */
X     if (outwind)
X     {
X       outwind = w_shrink(outwind);
X       outport = find_port(outwind);
X     }
X     break;
X
X   case BOTTOMTO:     /* send bottom window to top */
X     w_bottom();
X     break;
```

```
X
X    case WINSTYLE:     /* enable/disable sliders and arrows for new windows */
X      sliders = !sliders;
X      menu_icheck(menubar, WINSTYLE, sliders);
X      break;
X
X    case HIDEWIN: /* send top window to bottom */
X      if (outwind = w_hide())
X      {
X        outport = find_port(outwind);
X        printer_mark(outwind);
X      }
X      break;
X
X    case MOVEWIN:
X      if (outwind)
X      {
X        graf_mouse(POINT_HAND, &dummy);
X        evnt_button(1, 1, 1, &mx, &my, &dummy, &dummy);
X        wind_get(outwind, WF_CURRXYWH, &dummy, &dummy, &ww, &wh);
X        if (graf_dragbox(ww, wh, mx, my,
X          scr_x, scr_y, scr_w + ww, scr_h + wh, &mx, &my))
X            w_move(outwind, mx, my, ww, wh);
X        graf_mouse(sel_inp_mode? USER_DEF: ARROW, rmbmform);
X      }
X      break;
X
X    case OSIZEWIN:
X      if (outwind)
X      {
X        outwind = w_full(outwind);
X        outport = find_port(outwind);
X      }
X      break;
X
X    case CLOSEWIN:
X      if (outwind = proto_close(outwind))
X      {
X        printer_mark(outwind);
X        outport = find_port(outwind);
X      }
X      if (!uw_runs && !find_wind(1))
X      {
X        menu_tnormal(menubar, msgbuff[3], 1);
X        goto init;
X      }
X      break;
X
X    case WINTITLE:     /* enable/disable window headers */
X      titles = !titles;
X      menu_icheck(menubar, WINTITLE, titles);
X      break;
X
```

```
X    case SETCONF: /* set rs232 port configuration */
X      getrsconf();
X      if (s_dial(RSCONF, 3) == RCOK)
X        setrsconf();
X      break;
X
X    case WINCLEAR:     /* clear current window */
X      if (outwind)
X      {
X        w[outwind].ptr_status = LOG_NONE;
X        w_output(outwind, "\032");
X      }
X      break;
X    }
X    menu_tnormal(menubar, msgbuff[3], 1);
X    break;
X
X      case WM_NEWTOP:
X      case WM_TOPPED:
X    if (!locked)
X      w_top(msgbuff[3]);
X    break;
X
X      case WM_SIZED:
X      case WM_MOVED:
X    if (!locked)
X      w_move(msgbuff[3], msgbuff[4], msgbuff[5], msgbuff[6], msgbuff[7]);
X    break;
X
X      case WM_CLOSED:
X        if (locked) break;
X    if (cnt = proto_close(msgbuff[3]))
X    {
X      if (msgbuff[3] == outwind)     /* was keyboard input window closed? */
X        outwind = cnt;
X      printer_mark(outwind);
X      outport = find_port(outwind);
X    }
X    if (!uw_runs && !find_wind(1))
X    {
X      goto init;
X    }
X    break;
X
X      case WM_REDRAW:
X        if (highlighted_wdes == msgbuff[3])
X    {    /* this window has highlighted text, redraw all of window*/
X      register struct wi_str *wp = &w[msgbuff[3]];
X      w_redraw(msgbuff[3], FM_COPY, wp->x, wp->y, wp->w, wp->h);
X    }
X    else
X      /* redraw only damaged part */
X      w_redraw(msgbuff[3], FM_COPY, msgbuff[4], msgbuff[5], msgbuff[6],
```

```
msgbuff[7]);
X    break;
X
X      case WM_FULLED:
X    if (locked) break;
X    outwind = w_full(msgbuff[3]);
X    outport = find_port(outwind);
X    break;
X
X      case WM_ARROWED:
X    if (!locked)
X      w_arrow(msgbuff[3], msgbuff[4]);
X    break;
X
X      case WM_HSLID:
X      case WM_VSLID:
X    if (!locked)
X      w_slide(msgbuff[3], msgbuff[0] == WM_HSLID, msgbuff[4]);
X    break;
X      }
X    }
X    if ((event & MU_BUTTON) && ! locked)
X    {
X      if (sel_inp_mode && buttonstate) /* select input mode and button down */
X      {
X        int found;
X
X    graf_mouse(ARROW, &dummy);
X    if ((found = wind_find(mx, my)) != 0) {
X      outwind = found;
X      outport = find_port(outwind);
X      if (sel_inp_mode == PASTE)
X          proto_out(outport, pastebuff, strlen(pastebuff));
X    }
X    evnt_button(1, 2, 0, &dummy, &dummy, &dummy, &dummy);
X    buttonstate = 0;
X    sel_inp_mode = FALSE;
X      }
X      else if (my < scr_y) /* is mouse on menu bar? */
X      { /* yes, disable menu bar if button down */
X        if (buttonstate)
X    {
X          objc_change(menubar, DESK, 0, 0, 0, 0, 0,
X            menonoff? DISABLED: NORMAL, menonoff);
X          menu_bar(menubar, menonoff = !menonoff);
X      if (menonoff)
X        graf_mouse(ARROW, &dummy);
X      else
X        graf_mouse(USER_DEF, rmbmform);
X      evnt_button(1, 2, 0, &dummy, &dummy, &dummy, &dummy);
X      buttonstate = 0;
X    }
X      }
```

```
X      else   /* button click in work area */
X      { /* handle text selection or output or mouse packet */
X        int found, found1, x1, y1, x2, y2, w_x, w_y, w_w, w_h;
X
X    if ((found = wind_find(mx, my)) != 0)
X    {
X      wind_get(found, WF_WORKXYWH, &w_x, &w_y, &w_w, &w_h);
X      if (mx >= w_x && mx < w_x + w_w && my >= w_y && my <w_y + w_h)
X      { /* in window, fill paste buffer */
X        if (clicks > 1 && buttonstate)
X        {
X          y1 = (my - w_y - Y0) / w[found].font->inc_y;
X          copy_text(found, 0, y1, w[found].x_chrs - 1, y1, pastebuff);
X          evnt_button(3, 2, 0, &dummy, &dummy, &dummy, &dummy);
X          regionflag = 0;
X        }
X        else if (buttonstate)   /* if button down */
X        {
X          x1 = (mx - w_x - X0) / w[found].font->inc_x;
X          y1 = (my - w_y - Y0) / w[found].font->inc_y;
X          regionflag = 1;
X          found1 = found;
X        }
X        else if (regionflag && found == found1)
X        {    /* button up */
X          x2 = (mx - w_x - X0) / w[found].font->inc_x;
X          y2 = (my - w_y - Y0) / w[found].font->inc_y;
X          if (y2 == y1 && x2 == x1)
X          {   /* simple click on char; select word */
X            copy_word(found, x1, y1, pastebuff);
X          }
X          else if (y2 > y1 || (y2 == y1 && x2 > x1))
X            copy_text(found, x1, y1, x2-1, y2, pastebuff);
X          else
X         copy_text(found, x2+1, y2, x1, y1, pastebuff);
X        }
X      }
X      else
X      { /* in border */
X        /* send paste buffer and select keyboard window*/
X        if (buttonstate)   /* if button down */
X        {
X          char * pptr = pastebuff;
X          outwind = found;
X          outport = find_port(outwind);
X          if (clicks > 1)
X              proto_out(outport, pptr, strlen(pptr));
X        }
X        regionflag = 0;
X      }
X    }
X    else
X    {   /* not in window or border */
```

```
X        /* clear paste buffer and erase any current marks */
X      pastebuff[0] = '\0';
X      regionflag = 0;
X      copy_text(1, 1, 1, 0, 0, pastebuff);
X    }
X      }
X      buttonstate = buttonstate? 0: 2;   /* toggle buttonstate */
X    }
X    if (event & MU_M1)/* mouse moved into or out of work area */
X    {
X      if (!sel_inp_mode && !menonoff && !locked)
X    graf_mouse(m1inout? USER_DEF: ARROW, rmbmform);
X      m1inout = ! m1inout;
X    }
X  }
X}
X/*
X * printer_mark(wnd) places check marks in the apropriate places in the
X * printer menu.
X */
Xprinter_mark (wnd)
X{
X    objc_change(menubar, PRTBOTOM, 0, 0, 0, 0, 0,
X      (w[wnd].ptr_status & LOG_BOTOM)? CHECKED: NONE, 0);
X
X    objc_change(menubar, PRTTOP, 0, 0, 0, 0, 0,
X      (w[wnd].ptr_status & LOG_TOP)? CHECKED: NONE, 0);
X}
X
X/*
X * s_dial performs a simple dialog with buttons and text only.
X * The index of the terminating button is returned.
X * If action == 1, the dialog is displayed.  If action == 2, it is
X * removed.  If action == 3, both operations are done and form_do is
X * called.
X */
Xint s_dial(tree, action)
Xint tree, action;
X{
X  int tmp = 0;
X  int cx, cy, cw, ch;
X  OBJECT *obj_tmp;
X
X  rsrc_gaddr(R_TREE, tree, &obj_tmp);
X  form_center(obj_tmp, &cx, &cy, &cw, &ch);
X  if (action & 1) {
X    form_dial(FMD_START, 0, 0, 20, 10, cx, cy, cw, ch);
X    if (!fast) form_dial(FMD_GROW, 0, 0, 20, 10, cx, cy, cw, ch);
X    objc_draw(obj_tmp, 0, 5, cx, cy, cw, ch);
X  }
X  if (action == 3) {
X    tmp = form_do(obj_tmp, 0);
X  }
```

```
X  if (action & 2) {
X    if (!fast) form_dial(FMD_SHRINK, 0, 0, 20, 10, cx, cy, cw, ch);
X    form_dial(FMD_FINISH, 0, 0, 20, 10, cx, cy, cw, ch);
X    objc_change(obj_tmp, tmp, 0, cx, cy, cw, ch, NONE, 0);
X  }
X    return (tmp);
X}
X
X/*
X * set_menu_string sets the menu string for the specified menu object to
X * newstr.
X */
Xset_menu_string(newstr, object)
Xregister char * newstr;
Xint object;
X{
X  register char * oldstr;
X
X  oldstr = (char *) menubar[object].ob_spec + 2;
X  while (*oldstr && *newstr)
X    *oldstr++ = *newstr++;
X  if (*oldstr)
X   *oldstr = ' ';
X}
X
Xsize_dial()
X{
X  /*
X   * Enter rows and columns dialog
X   */
X  int cx, cy, cw, ch;
X  char *rowstr, *colstr;
X  rsrc_gaddr(R_TREE, WINDSIZE, &obj_tmp);
X  ted_tmp = (TEDINFO *) obj_tmp[WINDROWS].ob_spec;
X  rowstr = ((char *)ted_tmp->te_ptext);
X  if (atoi(rowstr) < 2)
X    strcpy (rowstr, "24");
X  ted_tmp = (TEDINFO *) obj_tmp[WINDCOLS].ob_spec;
X  colstr = ((char *)ted_tmp->te_ptext);
X  if (atoi(colstr) < 2)
X    strcpy (colstr, "80");
X  form_center(obj_tmp, &cx, &cy, &cw, &ch);
X  form_dial(FMD_START, 0, 0, 20, 10, cx, cy, cw, ch);
X  if (!fast)
X    form_dial(FMD_GROW, 0, 0, 20, 10, cx, cy, cw, ch);
X  objc_draw(obj_tmp, 0, 5, cx, cy, cw, ch);
X  tmp = form_do(obj_tmp, WINDROWS);
X  if (!fast)
X    form_dial(FMD_SHRINK, 0, 0, 20, 10, cx, cy, cw, ch);
X  form_dial(FMD_FINISH, 0, 0, 20, 10, cx, cy, cw, ch);
X  objc_change(obj_tmp, tmp, 0, cx, cy, cw, ch, NONE, 0);
X  if (tmp == WINDCANC)
X    return(FALSE);
```

```
X  xsiz = atoi(colstr);
X  ysiz = atoi(rowstr);
X  if (xsiz < 2)
X    xsiz = 2;
X  if (xsiz > 300)
X    xsiz = 300;
X  if (ysiz < 2)
X    ysiz = 2;
X  if (ysiz > 300)
X    ysiz = 300;
X  return (TRUE);
X}
SHAR_EOF
chmod 0600 WINMAIN.C || echo "restore of WINMAIN.C fails"
sed 's/^X//' << 'SHAR_EOF' > WINPROC.C &&
X/*
X * This file contains subroutines which deal with other processes
X */
X#include <obdefs.h>
X#include <gemdefs.h>
X#include <stdio.h>
X#include <osbind.h>
X#include <xbios.h>
X#include "wind.h"
X#include "uw.h"
X#include "windefs.h"
X
Xextern  char * environ;
X
Xextern struct wi_str w[];
Xextern int fast;
Xextern int   scr_x, scr_y, scr_w, scr_h; /* size of screen */
Xextern OBJECT    *menubar;
X
Xchar cmdpath[40] = "e:\\bin\\*.*";
X                      /* Path for command execution */
Xchar cmdname[40] = "msh.prg";       /* Name of command to run */
Xchar cmdargs[40] = " ";             /* Arguments for command */
X
X/*
X * Exec process from dialog.
X */
Xint do_exec()
X{
X  int status = 0;
X  int confbutt;
X
X  fsel_input(cmdpath, cmdname, &confbutt);
X  if (confbutt) {
X    extern char * rindex();
X    char cmdstr[80];
X    char cmdargv[40];
X    char *argv[20];
```

```
X    char * ind;
X    OBJECT *obj_tmp;
X    TEDINFO *ted_tmp;
X    int cx, cy, cw, ch, tmp;
X
X    strcpy(cmdstr, cmdpath);
X      ind = rindex(cmdstr, '\\');
X      if (ind) * ++ind = '\0';
X    strcat(cmdstr, cmdname);
X    form_dial(FMD_START, 0, 0, 0, 0, scr_x, scr_y, scr_w, scr_h);
X                        /* save screen */
X
X    rsrc_gaddr(R_TREE, PARAM, &obj_tmp);
X    ted_tmp = (TEDINFO *) obj_tmp[PARAMSTR].ob_spec;
X    strcpy(ted_tmp->te_ptext, cmdargs);
X    form_center(obj_tmp, &cx, &cy, &cw, &ch);
X    if (!fast) form_dial(FMD_GROW, 0, 0, 20, 10, cx, cy, cw, ch);
X    objc_draw(obj_tmp, 0, 5, cx, cy, cw, ch);
X    tmp = form_do(obj_tmp, PARAMSTR);
X    if (!fast) form_dial(FMD_SHRINK, 0, 0, 20, 10, cx, cy, cw, ch);
X    objc_change(obj_tmp, tmp, 0, cx, cy, cw, ch, NONE, 0);
X    if (tmp != OKEXEC) {
X      form_dial(FMD_FINISH, 0, 0, 0, 0, scr_x, scr_y, scr_w, scr_h);
X      return(0);
X    }
X    strcpy(cmdargs, ted_tmp->te_ptext);
X    strcpy(cmdargv, cmdargs);
X
X    graf_mouse(M_OFF, NULL);    /* turn mouse off */
X    menu_bar(menubar, 0);  /* menu bar off */
X    Cconws("\033E\033e");  /* clear screen, cursor on */
X    ind = cmdargv;
X    argv[0] = cmdname;
X    tmp = 1;
X    while (*ind != '\0') {
X      while (*ind == ' ' && *ind != '\0'){
X        *ind = '\0';
X        ++ind;
X      }
X      argv[tmp++] = ind;
X      while (*ind != ' ' && *ind != '\0')
X        ++ind;
X    }
X    argv[tmp] = NULL;
X    status = execve(cmdstr, argv, environ);
X    sleep(1);
X    Cconws("\033E\033f");  /* clear screen, cursor off */
X    form_dial(FMD_FINISH, 0, 0, 0, 0, scr_x, scr_y, scr_w, scr_h);
X    menu_bar(menubar, 1);  /* menu bar on */
X    graf_mouse(M_ON, NULL);     /* turn mouse on */
X  }
X  return(status);
X}
```

```
X
X/*
X * Set current path with dialog.
X */
Xint do_path()
X{
X  int status = 0;
X  int confbutt;
X  char curpath[80];
X  char ignore[40] = "";
X  int drv;
X
X  curpath[0] = Dgetdrv() + 'a';
X  curpath[1] = ':';
X  Dgetpath(curpath+2, 0);
X  strcat(curpath, "\\*.*");
X  fsel_input(curpath, ignore, &confbutt);
X  if (confbutt) {
X    extern char * index(), *rindex();
X    char * ind;
X    ind = index(curpath, ':');
X    if (ind) {
X      drv = *(ind - 1);
X      if (drv > '\\')
X        drv -= 'a';
X      else
X    drv -= 'A';
X      if (drv >= 0 && drv <= 15)
X        Dsetdrv(drv);
X    }
X    else
X      ind = curpath;
X    *rindex(ind, '\\') = 0;
X    status = Dsetpath(++ind);
X  }
X  return(status);
X}
SHAR_EOF
chmod 0600 WINPROC.C || echo "restore of WINPROC.C fails"
sed 's/^X//' << 'SHAR_EOF' > WINPROTO.C &&
X/*
X * This file contains the routines which implement the uw protocol.
X * Code has been migrating here from winmain.c, but its not all here yet.
X */
X
X#include <stdio.h>
X#include <osbind.h>
X#include <gemdefs.h>
X#include "wind.h"
X#include "uw.h"
X#include "windefs.h"
X
X#define LIMIT    300
```

```
X
Xextern  struct   wi_str   w[];
Xextern int outport;            /* ports used with uw */
Xextern int inwind, outwind;         /* windows for input/output */
Xextern int uw_runs;            /* is uw running ? */
XFNT *curfont;          /* font in use */
X
X/* proto_out sends length chars from str to the window defined by outport. */
Xproto_out(outport, str, length)
Xint outport, length;
Xchar *str;
X
X{
Xstatic int oldoutport = 1;
Xint key, i;
Xint outwind;
X
Xoutwind = find_wind(outport);
X  if (w[outwind].w_local == TRUE)    /* local window? */
X  {
X    char *found, *index();
X    char strcr[3] = "\r\n";
X
X    str[length] = '\0';
X    while (found = index(str, '\r'))
X    {
X      char line[500];
X      strncpy(line, str, found - str);
X      line[found-str] = '\0';
X      w_output(outwind, line);
X      w_output(outwind, strcr);
X      str = found + 1;
X    }
X    w_output(outwind, str);
X    return;
X  }
X  for (key = *str, i = 0; i < length; key = str[++i])
X  {
X    if (uw_runs)
X    {
X      if (outport != oldoutport)
X      {
X        xmitcmd(CB_FN_ISELW|outport);
X        oldoutport = outport;
X      }
X      if (key & 0x80) {
X        key &= 0x7f;
X    xmitcmd(CB_FN_META);
X      }
X      switch (key)
X      {
X      case XON:
X        xmitcmd(CB_FN_CTLCH|CB_CC_ON);
```

```
X        break;
X
X      case XOFF:
X        xmitcmd(CB_FN_CTLCH|CB_CC_OFF);
X        break;
X
X      case IAC:
X        xmitcmd(CB_FN_CTLCH|CB_CC_IAC);
X        break;
X
X      default:
X        Bconout(1, key);
X        break;
X      }
X    } else
X      Bconout(1, key);/* need mask with 0x7f? */
X  }
X}
X
X/*
X * proto_close closes the current window and notifies the remote.  The new
X * current window is returned.
X */
Xint proto_close(curwind)
Xint curwind;
X{
X  long dummy;
X
X  if (uw_runs && (find_port(curwind) < MAX_WIND))
X  {
X    xmitcmd(CB_FN_KILLW|find_port(curwind));
X  }
X  w_close(curwind);
X  wind_get(0, WF_TOP, &curwind, &dummy, &dummy, &dummy);
X  return (curwind);
X/*  } else
X  {
X    w_close(curwind);
X    return (0);
X  } */
X}
X
X/* proto_in receives characters from the serial port. */
Xproto_in()
X#define NORMSTATE 0
X#define IACSTATE 1
X/* #define METASTATE 2  replaced with seenmeta flag. */
X#define INITSTATE 3
X{
X  register char   *ptr;
X  char  str[LIMIT+2];
X  register char   chr;
X  register int    cnt;
```

```
X  long dummy;
X  static int state;
X  static int seenmeta = 0;
X
X      ptr = str;
X      cnt = 0;
X      while (Bconstat(1) && cnt++<LIMIT)
X      {
X    chr = Bconin(1)&0x7f;
X    switch (state)
X    {
X    case NORMSTATE:
X      if (!chr || chr=='\177') continue;
X      if (chr == IAC)
X      {
X        if (uw_runs)
X          state = IACSTATE;
X        else
X          state = INITSTATE;
X        continue;
X      }
X      if (seenmeta) {
X        seenmeta = 0;
X        *ptr++ = chr|0x80;
X      }
X      else *ptr++ = chr;
X      break;
X    case INITSTATE:
X      if (chr == CB_FN_MAINT | CB_MF_ENTRY){
X        xmitcmd(CB_FN_MAINT|CB_MF_ENTRY);
X        uw_runs = 1;
X        outport = 0;
X        outwind = 0;
X      }
X      else {
X        *ptr++ = IAC;
X        *ptr++ = chr;
X      }
X      state = NORMSTATE;
X      break;
X    case IACSTATE:
X      state = NORMSTATE;
X      if (chr&CB_DIR_MTOH) continue;
X      switch (chr&CB_FN)
X      {
X      case CB_FN_NEWW:
X        outport = w_open(chr&CB_WINDOW, "Terminal", curfont->def_win_x,
X          curfont->def_win_y);
X        outwind = find_wind(outport);
X        xmitcmd(CB_FN_ISELW|outport);
X        break;
X
X      case CB_FN_KILLW:
```

```
X       w_close(find_wind(chr&CB_WINDOW));
X       wind_get(0, WF_TOP, &outwind, &dummy, &dummy, &dummy);
X       outport = find_port(outwind);
X       break;
X
X     case CB_FN_OSELW:
X       if (inwind != find_wind(chr&CB_WINDOW))
X       {
X         *ptr = '\0';
X         if (ptr != str)
X         {
X           if (w[inwind].kerm_act)
X          rpack(" ", NULL, str);
X        else
X           w_output(inwind, str);
X         }
X         ptr = str;
X       }
X       inwind = find_wind(chr&CB_WINDOW);
X       break;
X
X     case CB_FN_META:
X       seenmeta = 1;
X       break;
X
X     case CB_FN_CTLCH:
X       switch (chr&CB_CC)
X       {
X       case CB_CC_IAC:
X         if (seenmeta) {
X           seenmeta = 0;
X           *ptr++ = IAC|0x80;
X         }
X         else *ptr++ = IAC;
X         break;
X
X       case CB_CC_ON:
X         if (seenmeta) {
X           seenmeta = 0;
X           *ptr++ = XON|0x80;
X         }
X         else *ptr++ = XON;
X         break;
X
X       case CB_CC_OFF:
X         if (seenmeta) {
X           seenmeta = 0;
X           *ptr++ = XOFF|0x80;
X         }
X         else *ptr++ = XOFF;
X
X         break;
X       }
```

```
X        break;
X
X      case CB_FN_MAINT:
X        switch (chr&CB_MF)
X        {
X        case CB_MF_ENTRY:
X          xmitcmd(CB_FN_MAINT|CB_MF_ENTRY);
X          uw_runs = 1;
X          outport = 0;
X          outwind = 0;
X          break;
X
X        case CB_MF_EXIT:
X          uw_runs = 0;
X          return (-1);
X        }
X        break;
X      }
X      break;
X    }
X      }
X      *ptr = '\0';
X      if (ptr!=str)
X      {
X        if (w[inwind].kerm_act)
X      rpack(" ", NULL, str);
X    else
X      w_output(inwind, str);
X      }
X      return (0);
X}
SHAR_EOF
chmod 0600 WINPROTO.C || echo "restore of WINPROTO.C fails"
sed 's/^X//' << 'SHAR_EOF' > WINSUBR.C &&
X/* subroutines for multi-window terminal emulation
X */
X
X#include <obdefs.h>
X#include <gemdefs.h>
X#include <osbind.h>
X#include <stdio.h>
X#include "wind.h"
X#include "windefs.h"
X
Xextern int handle;
X
X/* variables used by various routines
X */
X
Xlong dummy;                /* dummy return variable */
Xextern  int  outwind, outport;  /* window selection */
Xextern  int  scr_x, scr_y;      /* size of the screen */
Xextern  int  scr_w, scr_h;
```

```
Xextern  int  fast;              /* flag for fast open/close */
Xextern  int  overstrike;
Xextern  int  sliders;      /* flag for sliders on new windows */
Xextern  int  titles;            /* flag for title bars on new windows */
Xint tmp;               /* temporary for anything... */
Xextern  char alert[300];        /* used for alerts */
Xextern  FNT  *curfont;     /* current font */
Xextern  MFDB screen_mf;         /* screen descriptor */
Xextern  int  mouse;             /* for mouse on/off */
Xextern  int  audibell;     /* What happens on BEL? */
Xextern  int  visibell;
Xextern  int  toponbel;
X
Xstruct  wi_str   w[MAX_WIND];
X
X/* the program code...
X */
X
Xchar *getmem(size)
Xregister long size;
X{
X  char *got;
X
X  got = (char *) Malloc(size);
X#ifdef DEBUG
X  printf("alloc returned %lx of %ld size\n", got, size);
X#endif
X  if (got == NULL)
X  {
X    sprintf(alert, "[1][Could not get %ld bytes][Ok]", size);
X    form_alert(1, alert);
X  } else
X  {
X    bzero(got, size);
X  }
X  return got;
X}
X
Xbzero(ptr, size)
Xregister char *ptr;
Xregister long size;
X{
X  while (size--)
X    *ptr++ = 0;
X}
X
X/* find_port maps from window handles to UW port identifiers
X */
X
Xfind_port(wnd)
X{
X  return w[wnd].port;
X}
```

```
X
X/* find_wind maps from port ids to window handles
X */
X
Xfind_wind(port)
X{
Xint i;
X
X  for (i=1; i<MAX_WIND; i++)
X    if (w[i].port == port) return i;
X  return 0;
X}
X
X/* w_open opens a window with the supplied name and port and returns a port
X * number.  If port was zero, a free port is chosen.
X */
X
Xw_open(port, name, xsiz, ysiz)
Xchar *name;
X{
X  register struct wi_str *wp;
X  int wdes;
X  int i, cnt, wtyp;
X  int tmp_x, tmp_y, tmp_w, tmp_h;
X
X  if (port && find_wind(port)) return port;  /* this window is already open */
X  if (!port)
X  {
X    for (i=1; i<MAX_WIND; i++)
X    {
X      if (!find_wind(i))
X      {
X        port = i;     /* a free port */
X        break;
X      }
X    }
X  }
X
X  wtyp = (sliders ? WI_WITHSLD : 0) | (titles ? WI_NOSLD : 0);
X  wind_calc(0, wtyp, 0, 0, curfont->inc_x*xsiz+2*X0,
X    curfont->inc_y*ysiz+2*Y0, &dummy, &dummy, &tmp_w, &tmp_h);
X  if (tmp_w>scr_w)
X    tmp_w = scr_w;    /* full size <= screen size */
X  tmp_x = 10*(port-1);
X  if (tmp_h>scr_h)
X    tmp_h = scr_h;
X  tmp_y = scr_y+16*(port-1);
X
X  wdes = wind_create(wtyp, tmp_x, tmp_y, tmp_w,
X    tmp_h);
X  if (wdes < 0)
X  {
X    form_alert(1, "[1][Sorry, GEM has|no more windows|for us...][Ok]");
```

```
X    return 0;
X  }
X  wp = &w[wdes];
X  wp->wi_w = X0*2 + curfont->inc_x*xsiz;
X  wp->wi_h = Y0*2 + curfont->inc_y*ysiz;
X  wp->port = port;
X  if (!fast)
X    graf_growbox(0, 0, 20, 10, tmp_x, tmp_y, tmp_w, tmp_h);
X  wind_open(wdes, tmp_x, tmp_y, tmp_w, tmp_h);
X  wind_get(wdes, WF_WORKXYWH, &wp->x, &wp->y, &wp->w, &wp->h);
X  wp->fulled = 0;
X  wp->used = 1;
X  wp->x_off = 0;
X  wp->y_off = 0;
X  wp->px_off = 0;
X  wp->py_off = 0;
X  wp->m_off = wp->x & 15;
X  wp->cur_x = X0;
X  wp->cur_y = Y0;
X  wp->top_y = Y0;
X  wp->font = curfont;
X  wp->x_chrs = xsiz;
X  wp->y_chrs = ysiz;
X  wp->wi_mf.wpix = 2*X0 + xsiz*curfont->inc_x;
X  wp->wi_mf.hpix = 2*Y0 + ysiz*curfont->inc_y;
X#ifdef  KOPY
X  wp->wi_mf.wwords = ((wp->wi_mf.wpix>>5) +1) << 1;
X#else
X  wp->wi_mf.wwords = (wp->wi_mf.wpix>>4) +1;
X#endif
X  wp->wi_mf.ptr = getmem(((long)wp->wi_mf.hpix+wp->font->inc_y*MAXSCROLLED)
X                  *wp->wi_mf.wwords*2);
X  wp->wi_mf.format = 0;
X  wp->wi_mf.planes = 1;
X  wp->ptr_status = LOG_NONE;
X  wp->wi_style = wtyp;
X  w_rename(wdes, name);
X  strcpy(wp->wi_fpath, ".\\*.*");
X  wp->top_age = 1;
X  for (cnt = 1; cnt < MAX_WIND; cnt++)
X    if (w[cnt].port != 0)
X      w[cnt].top_age++;
X
X  setvslide(wdes);
X  sethslide(wdes);
X
X  return wp->port;
X}
X
X/* w_closei removes a window but does not release its storage.
X * This is used if the window contents must be saved for later use.
X */
X
```

```
Xw_closei(wdes)
X{
X  int xx, yy, ww, hh;
X  register struct wi_str *wp = &w[wdes];
X
X  if (!wp->used) return;
X  if (wp->wi_lfd)
X    fclose(wp->wi_lfd);
X  wind_get(wdes, WF_CURRXYWH, &xx, &yy, &ww, &hh);
X  wind_close(wdes);
X  if (!fast)
X    graf_shrinkbox(0, 0, 20, 10, xx, yy, ww, hh);
X  wind_delete(wdes);
X}
X
X/* w_close removes a window.  Most work is done by GEM, although w_close
X * does some cleanup functions, too.
X */
X
Xw_close(wdes)
X{
X  register struct wi_str *wp = &w[wdes];
X
X  if (!wp->used) return;
X  w_closei(wdes);
X  Mfree(wp->wi_mf.ptr);
X  bzero(wp, (long)sizeof (struct wi_str));
X}
X
X/* w_resize resizes an existing window.
X */
X
Xw_resize(wdes, xsiz, ysiz)
X{
X  register struct wi_str *wp1 = &w[wdes];
X  struct wi_str *wp2;
X  struct wi_str ws;
X  static int c[8];
X  int port, wind, i;
X
X  if (!wp1->used) return;
X  ws = *wp1;
X  port = find_port(wdes);
X  w_closei(wdes);
X  bzero(wp1, (long)sizeof (struct wi_str));
X  port = w_open(port, "", xsiz, ysiz);
X  wind = find_wind(port);
X  wp2 = &w[wind];
X  c[0] = ws.m_off;
X  c[1] = ws.top_y + max(0, ws.wi_mf.hpix - wp2->wi_mf.hpix);
X  c[2] = c[0] + min(ws.wi_mf.wpix, wp2->wi_mf.wpix);
X  c[3] = c[1] + min(ws.wi_mf.hpix, wp2->wi_mf.hpix);
X  c[4] = wp2->m_off;
```

```
X  c[5] = wp2->top_y;
X  c[6] = c[4] + min(ws.wi_mf.wpix, wp2->wi_mf.wpix);
X  c[7] = c[5] + min(ws.wi_mf.hpix, wp2->wi_mf.hpix);
X  /* copy screen */
X  vro_cpyfm(handle, FM_COPY, c, &ws.wi_mf, &wp2->wi_mf);
X  /* copy parameters */
X  wp2->inverse = ws.inverse;
X  wp2->insmode = ws.insmode;
X  if (wp2->font != ws.font)
X  {
X    wp2->cur_x = X0;
X    wp2->cur_y = (wp2->y_chrs - 1) * wp2->font->inc_y + Y0;
X  }
X  else
X  {
X    wp2->cur_x = (wp2->x_chrs - 1) * wp2->font->inc_x + X0;
X    if (ws.cur_x < wp2->cur_x)
X      wp2->cur_x = ws.cur_x;
X    wp2->cur_y = max(0, ws.cur_y - c[1]) + Y0;
X  }
X  wp2->state = ws.state;
X  for (i=0; i<80; i++) wp2->nuname[i] = ws.nuname[i];
X  for (i=0; i<80; i++) wp2->wi_fpath[i] = ws.wi_fpath[i];
X  for (i=0; i<20; i++) wp2->wi_fname[i] = ws.wi_fname[i];
X  wp2->nuptr = ws.nuptr;
X  wp2->ptr_status = ws.ptr_status;
X  wp2->wi_lfd = ws.wi_lfd;
X  wp2->kerm_act = ws.kerm_act;
X  w_rename(wind, ws.name);
X
X  Mfree(ws.wi_mf.ptr);
X  return port;
X}
X
X/* w_rename changes the title bar of a window
X */
X
Xw_rename(wdes, name)
Xchar *name;
X{
X  register struct wi_str *wp = &w[wdes];
X
X  if (name)
X    strcpy(wp->name, name);
X  if (wp->wi_style & NAME)
X  {
X    sprintf(wp->dname, " %s%s %s ", (wp->ptr_status != LOG_NONE)? "\275": "",
X      wp->wi_lfd? "\237": "", wp->name);
X    wind_set(wdes, WF_NAME, wp->dname + (wp->dname[1] == ' '), 0, 0);
X  }
X}
X
X/* w_redraw redraws part of the screen from window contents.
```

```
X * The coordinates are screen relative.
X */
X
Xw_redraw(wdes, logic, xx, yy, ww, hh)
X{
X  static int c[8];
X  static GRECT t1, t2;
X  register struct wi_str *wp = &w[wdes];
X
X  if (xx+ww > scr_w)
X    ww = scr_w - xx;
X  if (yy+hh > scr_h+scr_y)
X    hh = scr_h+scr_y - yy;
X  t2.g_x = xx; t2.g_y = yy;
X  t2.g_w = ww; t2.g_h = hh;
X  t1.g_x = wp->x; t1.g_y = wp->y;
X  t1.g_w = wp->w; t1.g_h = wp->h;
X  if (!rc_intersect(&t2, &t1)) return;   /* nothing to do... */
X  wind_update(TRUE);
X  wind_get(wdes, WF_FIRSTXYWH, &t1.g_x, &t1.g_y, &t1.g_w, &t1.g_h);
X  while (t1.g_w && t1.g_h)
X  {
X    if (rc_intersect(&t2, &t1))
X    {
X      if (mouse)
X      {
X    /* we have to do graphics, so switch the mouse off.
X     * mouse will be switched on again in main loop.
X     * this is ugly, but it improves speed a bit...
X     */
X    mouse = 0;
X    graf_mouse(M_OFF, NULL);
X      }
X#ifdef  KCOPY
X      tfbmr(wp->wi_mf.ptr, t1.g_x - wp->x + wp->x_off + wp->m_off,
X    t1.g_y - wp->y + wp->y_off + wp->top_y - Y0, wp->wi_mf.wwords >> 1,
X    screen_mf.ptr, t1.g_x, t1.g_y, screen_mf.wwords >> 1,
X    t1.g_w, t1.g_h, logic);
X#else
X      c[0] = t1.g_x - wp->x + wp->x_off + wp->m_off;
X      c[1] = t1.g_y - wp->y + wp->y_off + wp->top_y - Y0;
X      c[2] = c[0] + t1.g_w - 1;
X      c[3] = c[1] + t1.g_h - 1;
X      c[4] = t1.g_x;
X      c[5] = t1.g_y;
X      c[6] = c[4] + t1.g_w - 1;
X      c[7] = c[5] + t1.g_h - 1;
X      vro_cpyfm(handle, logic, c, &wp->wi_mf, &screen_mf);
X#endif
X    }
X    wind_get(wdes, WF_NEXTXYWH, &t1.g_x, &t1.g_y, &t1.g_w, &t1.g_h);
X  }
X  if (!fast && !mouse)
```

```
X  {
X    mouse = 1;
X    graf_mouse(M_ON, NULL);
X  }
X  wind_update(FALSE);
X}
X
X/* w_update copies a portion of the window to the screen.  Coordinates
X * are window-relative
X */
X
Xw_update(wdes, logic, xx, yy, ww, hh)
X{
X  register struct wi_str *wp = &w[wdes];
X
X  w_redraw(wdes, logic, xx + wp->x - wp->x_off,
X         yy + wp->y - wp->y_off - wp->top_y + Y0, ww, hh);
X}
X
X/* w_move sets the window's idea of its own position on the screen
X */
X
Xw_move(wdes, xx, yy, ww, hh)
X{
X  register struct wi_str *wp = &w[wdes];
X  int flag = 0;
X  int m_w, m_h;
X  int x1, x2;
X  int tmp;
X
X  wind_calc(1, wp->wi_style, xx, yy, ww, hh, &dummy, &dummy, &m_w, &m_h);
X  if (tmp = (m_w-2*X0)%wp->font->inc_x)
X  {
X    ww -= tmp;
X    m_w -= tmp;
X  }
X  if (tmp = (m_h-2*Y0)%wp->font->inc_y)
X  {
X    hh -= tmp;
X    m_h -= tmp;
X  }
X  if (m_w>wp->wi_w) ww = ww-(m_w-wp->wi_w);
X  if (m_h>wp->wi_h) hh = hh-(m_h-wp->wi_h);
X  wind_set(wdes, WF_CURRXYWH, xx, yy, ww, hh);
X  wind_get(wdes, WF_WORKXYWH, &wp->x, &wp->y, &wp->w, &wp->h);
X  if (wp->x_off+wp->w > wp->wi_w)
X  {
X    register int inc_x = wp->font->inc_x;
X    flag = 1;
X    wp->x_off = (wp->wi_w - wp->w)/inc_x*inc_x;
X  }
X  if (wp->y_off+wp->h > wp->wi_h)
X  {
```

```
X    register int inc_y = wp->font->inc_y;
X    flag = 1;
X    wp->y_off = (wp->wi_h - wp->h)/inc_y*inc_y;
X  }
X  x1 = wp->m_off;
X  x2 = (wp->x - wp->x_off) & 15;
X  if (x1 != x2)
X  {
X#ifdef  KOPY
X    int top = wp->top_y;
X    int cnt = wp->wi_mf.wwords >> 1;
X    cpbmr(wp->wi_mf.ptr, x1, top, cnt, wp->wi_mf.ptr, x2, top, cnt,
X      wp->wi_w, wp->wi_h);
X#else
X  int c[8];
X    c[0] = x1;
X    c[1] = wp->top_y;  /* displayed part of memory form starts here */
X    c[2] = x1 + wp->wi_w - 1;
X    c[3] = wp->wi_h - 1 + c[1];
X    c[4] = x2;
X    c[5] = c[1];
X    c[6] = x2 + wp->wi_w - 1;
X    c[7] = c[3];
X    vro_cpyfm(handle, 3, c, &wp->wi_mf, &wp->wi_mf);
X#endif
X    wp->m_off = x2;
X  }
X  if (flag)
X    w_redraw(wdes, FM_COPY, wp->x, wp->y, wp->w, wp->h);
X  setvslide(wdes);
X  sethslide(wdes);
X}
X
X/*
X * w_top makes win the top window.
X */
Xw_top(win)
Xint win;
X{
X  int cnt;
X
X  wind_set(win, WF_TOP, 0, 0, 0, 0);
X  outport = find_port(win);
X  outwind = win;
X  w[outwind].top_age = 0;
X  for (cnt = 1; cnt < MAX_WIND; cnt++)
X    if (find_port(cnt))    /* if window is in use */
X      w[cnt].top_age++;
X  printer_mark(outwind);
X}
X
X/*
X * w_bottom finds the bottom window and puts it on top
```

```
X */
Xw_bottom()
X{
X  int i;
X  int highwin, highcnt;
X
X  highcnt = 0;
X  highwin = 1;
X  for (i = 1; i < MAX_WIND; i++)
X    if (w[i].top_age > highcnt)
X    {
X      highcnt = w[i].top_age;
X      highwin = i;
X    }
X  if (w[highwin].used)
X    w_top(highwin);
X}
X
X/*
X * w_hide puts the top window on the bottom.  The new top window is returned.
X */
Xint w_hide()
X{
X  int i, j;
X  int newtop, botwin, highcnt;
X  int oldtop, dum;
X
X  wind_get(0, WF_TOP, &oldtop, &dum, &dum, &dum);
X  if (!oldtop)
X    return (0);
X  j = -1;
X  botwin = 0;
X  for (j = -1; ; j--)
X  {
X    newtop = botwin;
X    botwin = oldtop;
X    highcnt = 0;
X    for (i = 1; i < MAX_WIND; i++)
X    {
X      if (w[i].top_age > highcnt)
X      {
X        highcnt = w[i].top_age;
X        botwin = i;
X      }
X    }
X    if (botwin == oldtop)
X      break;
X    wind_set(botwin, WF_TOP, 0, 0, 0, 0);
X    w[botwin].top_age = j; /*top_age less than untouched windows (top_age < 0)*/
X  }
X  for (j = 1; j < MAX_WIND; j++)
X    if (find_port(j)) /* if window is in use */
X      w[j].top_age += MAX_WIND + 3; /* correct top_age to reflect new order */
```

```
X  return(newtop ? newtop : oldtop);
X}
X
X#define TINYX 80
X#define TINYY 70
X/*
X * w_shrink saves current size and location and shrinks to standard tiny size.
X * The second from the top non-shrunk window is placed on top.
X */
Xw_shrink(wdes)
Xint wdes;
X{
X  register struct wi_str *wp = &w[wdes];
X  int curw, dummy;
X
X  /*
X   * Don't shrink a window that is currently shrunk
X   */
X  wind_get(wdes, WF_CURRXYWH, &dummy, &dummy, &curw, &dummy);
X  if (curw <= TINYX)
X    return;
X
X  /*
X   * Set up fulled and previous size and location window variables to tiny
X   * size.  Then call w_full
X   */
X  wp->fulled = 1;
X  wp->px = scr_x + scr_w - TINYX + 2;
X  wp->py = scr_y + (wdes - 1) * (TINYY - 2);
X  wp->pw = TINYX;
X  wp->ph = TINYY;
X  if (wp->py + 10 > scr_y + scr_h)
X  {
X    wp->py = wp->py - scr_h + 10;
X    wp->px = scr_x + scr_w - TINYY * 2;
X  }
X
X  return (w_full(wdes));
X}
X
X/* w_full toggles size and location
X */
Xw_full(wdes)
X{
X  register struct wi_str *wp = &w[wdes];
X  int x1, y1, w1, h1;
X  int x2, y2, w2, h2;
X  int full, topwin;
X
X  full = wp->fulled;
X  if (full) {
X    x1 = wp->px;
X    y1 = wp->py;
```

```
X    w1 = wp->pw;
X    h1 = wp->ph;
X  } else
X    wind_get(wdes, WF_FULLXYWH, &x1, &y1, &w1, &h1);
X
X  wind_get(wdes, WF_CURRXYWH, &x2, &y2, &w2, &h2);
X  wp->px = x2;
X  wp->py = y2;
X  wp->pw = w2;
X  wp->ph = h2;
X  if (!fast)
X  {
X    if (w2>=w1 && h2>=h1)
X      graf_growbox(x2, y2, w2, h2, x1, y1, w1, h1);
X    else if (w2<=w1 && h2<=h1)
X      graf_shrinkbox(x1, y1, w1, h1, x2, y2, w2, h2);
X  }
X
X  x2 = wp->x_off;
X  y2 = wp->y_off;
X  wp->x_off = wp->px_off;
X  wp->y_off = wp->py_off;
X  wp->px_off = x2;
X  wp->py_off = y2;
X
X  w_move(wdes, x1, y1, w1, h1);
X  w_redraw(wdes, FM_COPY, wp->x, wp->y, wp->w, wp->h);
X  wp->fulled = 1;
X  topwin = wdes;
X  if (h1 < TINYY + 10)
X  {
X    int lowcnt, i;
X
X    lowcnt = 32000;
X    for (i = 1; i< MAX_WIND; i++)
X    {
X      if (find_port(i))
X      {
X        wind_get(i, WF_CURRXYWH, &x2, &y2, &w2, &h2);
X
X        if (w[i].top_age < lowcnt && h2 > TINYY + 10 )
X        { /* find top non-tiny open window */
X          lowcnt = w[i].top_age;
X          topwin = i;
X    }
X      }
X    }
X    wind_set(topwin, WF_TOP, 0, 0, 0, 0);
X  }
X  return(topwin);
X}
X
Xw_arrow(wdes, arrow)
```

```
X{
X  register struct wi_str *wp = &w[wdes];
X  int inc_x = wp->font->inc_x;
X  int inc_y = wp->font->inc_y;
X
X  switch (arrow)
X  {
X  case 0:    /* page up */
X    wp->y_off -= wp->h/inc_y*inc_y;
X    goto y_upd;
X
X  case 1:    /* page down */
X    wp->y_off += wp->h/inc_y*inc_y;
X    goto y_upd;
X
X  case 2:    /* row up */
X    wp->y_off -= inc_y;
X    goto y_upd;
X
X  case 3:    /* row down */
X    wp->y_off += inc_y;
X    goto y_upd;
X
X  case 4:    /* page left */
X    wp->x_off -= wp->w/inc_x*inc_x;
X    goto x_upd;
X
X  case 5:    /* page right */
X    wp->x_off += wp->w/inc_x*inc_x;
X    goto x_upd;
X
X  case 6:    /* column left */
X    wp->x_off -= inc_x;
X    goto x_upd;
X
X  case 7:    /* column right */
X    wp->x_off += inc_x;
X    goto x_upd;
X  }
X
Xx_upd:
X  if (wp->x_off<0) wp->x_off = 0; else
X  if (wp->x_off+wp->w > wp->wi_w) wp->x_off = (wp->wi_w - wp->w)/inc_x*inc_x;
X  sethslide(wdes);
X  goto upd;
X
Xy_upd:
X  if (wp->y_off<0) wp->y_off = 0; else
X  if (wp->y_off+wp->h > wp->wi_h) wp->y_off = (wp->wi_h - wp->h)/inc_y*inc_y;
X  setvslide(wdes);
X
Xupd:
X  w_redraw(wdes, FM_COPY, wp->x, wp->y, wp->w, wp->h);
```

```
X}
X
Xw_slide(wdes, hor, val)
X{
X  register struct wi_str *wp = &w[wdes];
X
X  if (hor)
X  {
X    tmp = wp->font->inc_x;
X    wp->x_off = ((long)val*(wp->wi_w-wp->w)/1000)/tmp*tmp;
X    sethslide(wdes);
X  } else
X  {
X    tmp = wp->font->inc_y;
X    wp->y_off = ((long)val*(wp->wi_h-wp->h)/1000)/tmp*tmp;
X    setvslide(wdes);
X  }
X  w_redraw(wdes, FM_COPY, wp->x, wp->y, wp->w, wp->h);
X}
X
Xsethslide(wdes)
X{
X  register struct wi_str *wp = &w[wdes];
X
X  if (wp->wi_style & HSLIDE)
X  {
X    tmp = (long)1000*wp->x_off/(wp->wi_w-wp->w);
X    wind_set(wdes, WF_HSLIDE, tmp, 0, 0, 0);
X    tmp = (long)1000*wp->w/wp->wi_w;
X    wind_set(wdes, WF_HSLSIZE, tmp, 0, 0, 0);
X  }
X}
X
Xsetvslide(wdes)
X{
X  register struct wi_str *wp = &w[wdes];
X
X  if (wp->wi_style & VSLIDE)
X  {
X    tmp = (long)1000*wp->y_off/(wp->wi_h-wp->h);
X    wind_set(wdes, WF_VSLIDE, tmp, 0, 0, 0);
X    tmp = (long)1000*wp->h/wp->wi_h;
X    wind_set(wdes, WF_VSLSIZE, tmp, 0, 0, 0);
X  }
X}
X
Xw_flash(wdes, state)
X{
X  register struct wi_str *wp = &w[wdes];
X  static int wdes_last;
X#ifdef  KOPY
X  int x, y, cnt;
X#else
```

```
X  int t[8];
X#endif
X
X  if (wdes != wdes_last) w_flash(wdes_last, 1);
X  wdes_last = wdes;
X  if (!wp->used || wp->curstate == state) return;
X  if (state == 2)
X    wp->curstate = !wp->curstate;
X  else
X    wp->curstate = state;
X#ifdef  KOPY
X  x = wp->cur_x + wp->m_off;
X  y = wp->cur_y;
X  cnt = wp->wi_mf.wwords >> 1;
X  tfbmr(wp->wi_mf.ptr, x, y, cnt, wp->wi_mf.ptr, x, y, cnt,
X    wp->font->inc_x, wp->font->inc_y, 12);
X#else
X  t[0] = t[4] = wp->cur_x + wp->m_off;
X  t[1] = t[5] = wp->cur_y;
X  t[2] = t[6] = t[0]+wp->font->inc_x-1;
X  t[3] = t[7] = t[1]+wp->font->inc_y-1;
X  vro_cpyfm(handle, 12, t, &wp->wi_mf, &wp->wi_mf);
X#endif
X  w_update(wdes, FM_COPY, wp->cur_x, wp->cur_y, wp->font->inc_x, wp->font-
>inc_y);
X}
X
X/* w_output prints a string onto the window.  The string may
X * contain control chars and escape sequences.  It ends with \0.
X */
X
Xw_output(wdes, ptr)
Xchar *ptr;
X{
X  register struct wi_str *wp = &w[wdes];
X  static char ch;
X  static int inc_x, cur_x;
X  static int state;
X  static int inc_y, cur_y;
X#ifdef  KOPY
X  int x, w, cnt;
X#else
X  static int t[8];
X#endif
X  static int f_x, f_y, f_mod;
X  static int scrolled; /* Number of scrolling operations delayed */
X  static int xsiz, ysiz;/* Size in chars of terminal emulation for this window*/
X  register unsigned char *sptr;
X  register unsigned long *dptr;
X  register unsigned long mask;
X  register int shift;
X  register unsigned long val;
X  static int count;
```

```
X  static char * fdata;
X  static long width;
X  static char * wimfptr;
X  static int moffincx;
X  static char * savptr;
X
X
X#ifdef DEBUG
X  printf("entered w_output\n");
X#endif
X  if (!wp->font) return;
X  state = wp->state;
X  inc_x = wp->font->inc_x;
X  inc_y = wp->font->inc_y;
X  xsiz = wp->x_chrs;
X  ysiz = wp->y_chrs;
X  f_x = cur_x = wp->cur_x;
X  f_y = cur_y = wp->cur_y;
X  scrolled = wp->top_y/inc_y;
X  fdata = wp->font->f_data;
X  width = 2 * wp->wi_mf.wwords;
X  wimfptr = ((char *) (wp-> wi_mf.ptr)) - 2;
X  moffincx = wp->m_off + inc_x - 1;
X  f_mod = 0;
X  savptr = ptr;
X
X  if (wp->curstate) w_flash(wdes, 0);
X
X  while (ch = *ptr++)
X  {
X    switch (state)
X    {
X    case S_NORMAL:
X      if (ch >= ' ')
X      {
X    if (wp->insmode) /* open space for character */
X    {
X#ifdef  KOPY
X      x = cur_x + wp->m_off;
X      cnt = wp->wi_mf.wwords >> 1;
X      cpbmr(wp->wi_mf.ptr, x, cur_y, cnt,
X        wp->wi_mf.ptr, x + inc_x, cur_y, cnt,
X        (xsiz - 1) * inc_x + wp->m_off - x, inc_y);
X#else
X      t[0] = cur_x + wp->m_off;
X      t[1] = t[5] = cur_y;
X      t[2] = (xsiz - 1) * inc_x + wp->m_off - 1;
X      t[3] = t[7] = cur_y + inc_y - 1;
X      t[4] = t[0] + inc_x;
X      t[6] = t[2] + inc_x;
X      vro_cpyfm(handle, 3, t, &wp->wi_mf, &wp->wi_mf);
X#endif
X    }
```

```
X    /* paint the character */
X    sptr = fdata+ch*16;
X    dptr = wimfptr + cur_y*width
X         + (((moffincx + cur_x)>>4)<<1);
X    shift = 15 - ((moffincx + cur_x)&15);
X    if (overstrike)
X      mask = -1L;
X    else
X      mask = (-1L<<(shift+inc_x)|(1<<shift)-1);
X    if (wp->inverse)
X    {
X      for (count = inc_y; count; count--)
X      {
X        val = ((long)(*sptr++))<<shift ^ ~mask;
X        *dptr = (*dptr&mask)|val;
X        ((char *)dptr) += width;
X      }
X    } else
X    {
X      for (count = inc_y; count; count--)
X      {
X        val = ((long)(*sptr++))<<shift;
X        *dptr = (*dptr&mask)|val;
X        ((char *)dptr) += width;
X      }
X    }
X        cur_x += inc_x;
X    f_mod = 1;
X    if (cur_x > inc_x * xsiz) /* autowrap */
X    {
X      cur_y += inc_y;
X      if (cur_y >= wp->top_y + inc_y * ysiz) {
X        if (wp->ptr_status & LOG_TOP)
X          dump_line(wdes, 0);
X        wp->top_y += inc_y;
X        ++ scrolled;
X      }
X      if (! scrolled)
X      {
X        w_update(wdes, FM_COPY, f_x, f_y, cur_x-f_x, inc_y);
X        f_mod = 0;
X      }
X      cur_x = X0;
X
X      f_x = cur_x;
X      f_y = cur_y;
X    }
X      } else /* not printable character */
X      {
X    if (f_mod && !scrolled)
X    {
X      if (!wp->insmode)
X        w_update(wdes, FM_COPY, f_x, f_y, cur_x - f_x, inc_y);
```

```
X      else
X        w_update(wdes, FM_COPY, f_x, f_y, xsiz * inc_x-f_x, inc_y);
X      f_mod = 0;
X    }
X    switch (ch)
X    {
X    case '\007':  /* Bell */
X      if (audibell)
X        Bconout(2, '\007');
X      if (toponbel)
X        /* put this window on top but don't make it the current window.
X         * should we set a flag so that the first input from the current
X         * window puts it back on top?
X         */
X        wind_set(wdes, WF_TOP, wdes, 0, 0, 0);
X      if (visibell)
X      {
X        w_redraw(wdes, FM_INVERT, wp->x, wp->y, wp->w, wp->h);
X        w_redraw(wdes, FM_COPY, wp->x, wp->y, wp->w, wp->h);
X        /* Should clear flag to prevent need for next update? */
X      }
X      break;
X
X    case '\r':    /* Carriage Return */
X      cur_x = X0;
X      break;
X
X    case '\b':    /* Backspace */
X      if (cur_x>X0) cur_x -= inc_x;
X      break;
X
X    case '\n':    /* Newline */
X      cur_y += inc_y;
X      if (cur_y >= inc_y * ysiz + wp->top_y)
X      {
X        if (wp->ptr_status & LOG_TOP)
X          dump_line(wdes, 0);
X        wp->top_y += inc_y;
SHAR_EOF
echo "End of part 4, continue with part 5"
echo "5" > s2_seq_.tmp
exit 0


From 7776767858772205262
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,1dd6e6b90a8c1c57,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-02 14:07:04 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!csd4.milw.wisc.edu!bionet!agate!
saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
```

Subject: v02i023: uw -- Unix Windows windowing terminal software part05/05
Keywords: shar
Message-ID: <6555@saturn.ucsc.edu>
Date: 2 Mar 89 22:07:04 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 379
Approved: koreth@ssyx.ucsc.edu

Submitted-by: sun.com!laidbak!katzung (Brian Katzung)
Posting-number: Volume 2, Issue 23
Archive-name: uw/part05

```
#!/bin/sh
# this is part 5 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file WINSUBR.C continued
#
CurArch=5
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> WINSUBR.C
X       ++ scrolled;
X     }
X     break;
X
X   case '\036':  /* Home */
X     cur_x = X0;
X     cur_y = wp->top_y;
X     break;
X
X   case '\013':  /* Cursor Up */
X     if (cur_y!=wp->top_y) {
X       cur_y -= inc_y;
X     }
X     break;
X
X   case '\014':  /* Cursor Right */
X     if (cur_x < (xsiz - 1) * inc_x) cur_x += inc_x;
X     break;
X
X   case '\032':  /* Clear Screen */
X     wp->inverse = wp->insmode = 0;
X     cur_x = X0;
X     cur_y = Y0;
X     wp->top_y = Y0;
X     lineerase(wp, 0, ysiz - 1 + MAXSCROLLED);
```

```
X      ++ scrolled;
X      break;
X
X    case '\t':    /* Tab */
X      cur_x = ((cur_x/inc_x/8+1))*inc_x*8+X0;
X      break;
X
X    case '\033':  /* ESC */
X      state = S_ESC;
X      count = 0; /* count is used for insert or delete line */
X      break;
X    }
X    f_x = cur_x;
X    f_y = cur_y;
X      }
X      break;
X
X     case S_ESC:
X      if (ch >= '0' && ch <= '9') {
X        count = count * 10 + ch - '0';
X    break;
X      }
X      switch (ch)
X      {
X      case '*':       /* Clear Screen */
X    f_x = cur_x = X0;
X    wp->top_y = f_y = cur_y = Y0;
X    wp->inverse = wp->insmode = 0;
X    lineerase(wp, 0, ysiz - 1 + MAXSCROLLED);
X    ++ scrolled;
X    state = S_NORMAL;
X    break;
X
X      case 'Y':       /* Clear to End of Screen */
X    lineerase(wp, cur_y / inc_y + 1, ysiz - 1 + wp->top_y / inc_y);
X    if (! scrolled)
X      w_update(wdes, FM_COPY, X0, cur_y + inc_y, xsiz*inc_x, ysiz*inc_y);
X    /* fall through */
X      case 'T':       /* Clear to End of Line */
X#ifdef  KOPY
X    x = cur_x + wp->m_off;
X    cnt = wp->wi_mf.wwords >> 1;
X    tfbmr((short *) 0, 0, 0, 0, wp->wi_mf.ptr, x, cur_y, cnt,
X      X0 + xsiz * inc_x + wp->m_off - x, inc_y, 0);
X#else
X    t[0] = t[4] = cur_x + wp->m_off;
X    t[1] = t[5] = cur_y;
X    t[2] = t[6] = X0-1 + xsiz*inc_x +wp->m_off;
X    t[3] = t[7] = cur_y+inc_y-1;
X    vro_cpyfm(handle, 0, t, &wp->wi_mf, &wp->wi_mf);
X#endif
X    if (! scrolled)
X      w_update(wdes, FM_COPY, cur_x, cur_y, xsiz * inc_x - cur_x, inc_y);
```

```
X    state = S_NORMAL;
X    break;
X
X      case 'E':       /* Add Line */
X      case 'R':       /* Delete Line */
X        if (count == 0) {
X      count = 1;
X      /* Look ahead for contiguous insert/delete line operations */
X      while (*ptr == '\033' && ptr[1] == ch) {
X        ptr +=2;
X        count ++;
X      }
X    }
X    if (ch == 'E')
X      scrolldn(wp, cur_y/inc_y, ysiz-(cur_y-wp->top_y+Y0)/inc_y-count, count);
X    else
X      scrollup(wp, cur_y / inc_y,
X        ysiz - (cur_y - wp->top_y + Y0)/inc_y - count, count);
X    if (! scrolled)
X      w_update(wdes, FM_COPY, X0, cur_y, xsiz * inc_x,
X        ysiz * inc_y - cur_y + wp->top_y - Y0);
X    state = S_NORMAL;
X    break;
X
X      case 'q':       /* Insert Mode */
X    wp->insmode = 1;
X    state = S_NORMAL;
X    break;
X
X      case 'r':       /* End Insert */
X    wp->insmode = 0;
X    state = S_NORMAL;
X    break;
X
X      case 'W':       /* Delete Character */
X#ifdef  KOPY
X    x = cur_x + wp->m_off;
X    w = X0 + xsiz * inc_x + wp->m_off - x;
X    cnt = wp->wi_mf.wwords >> 1;
X    cpbmr(wp->wi_mf.ptr, x + inc_x, cur_y, cnt,
X      wp->wi_mf.ptr, x, cur_y, cnt, w, inc_y);
X    tfbmr((short *) 0, 0, 0, 0,
X      wp->wi_mf.ptr, X0 + (xsiz - 1) * inc_x + wp->m_off, cur_y, cnt,
X      inc_x, inc_y, 0);
X#else
X    t[0] = cur_x + inc_x + wp->m_off;
X    t[1] = t[5] = cur_y;
X    t[2] = X0 - 1 + xsiz * inc_x + wp->m_off;
X    t[3] = t[7] = cur_y+inc_y - 1;
X    t[4] = t[0] - inc_x;
X    t[6] = t[2] - inc_x;
X    vro_cpyfm(handle, 3, t, &wp->wi_mf, &wp->wi_mf);
X    t[0] = t[4] = X0 + (xsiz - 1) * inc_x + wp->m_off;
```

```
X    t[2] = t[6] = t[0] + inc_x - 1;
X    vro_cpyfm(handle, 0, t, &wp->wi_mf, &wp->wi_mf);
X#endif
X    if (! scrolled)
X      w_update(wdes, FM_COPY, cur_x, cur_y, xsiz * inc_x - (cur_x - X0),
X        inc_y);
X    state = S_NORMAL;
X    break;
X
X      case '=':       /* Cursor Movement */
X    state = S_ESC1;
X    break;
X
X      case 'G':       /* Switch Inverse/Normal */
X    state = S_ESC2;
X    break;
X
X      case 'S':       /* Change Status Line */
X    state = S_STATUS;
X    wp->nuptr = 0;
X    break;
X
X      case '[':       /* ascii escape */
X        state = S_ESCA;
X    wp->nuptr = 0;
X    break;
X
X      default:        /* Unknown escape sequence */
X    state = S_NORMAL;
X      }
X      break;
X
X    case S_ESC1:  /* get line number */
X      if (ch < ' ' || ch >= ' ' + ysiz) ch = ' ';
X      f_y = cur_y = (ch-' ')*inc_y + wp->top_y;
X      state = S_ESC3;
X      break;
X
X    case S_ESC3:  /* get column number */
X      if (ch < ' ' || ch >= ' ' + xsiz) ch = ' ';
X      f_x = cur_x = (ch-' ')*inc_x +X0;
X      state = S_NORMAL;
X      break;
X
X    case S_ESC2:  /* get normal/inverse state */
X      wp->inverse = ch&1;
X      state = S_NORMAL;
X      break;
X
X    case S_STATUS:
X      if (ch == '\r')
X      {
X    wp->nuname[wp->nuptr] = '\0';
```

```
X    w_rename(wdes, wp->nuname);
X    state = S_NORMAL;
X      } else if (wp->nuptr < 72)
X      {
X    wp->nuname[wp->nuptr++] = ch;
X      }
X      break;
X    case S_ESCA:
X      if (ch >= '0' && ch <= '9')
X        wp->nuptr = ch - '0' + 10 * wp->nuptr;
X      else if (ch == 't')
X      {
X        if (wp-> nuptr == 18)
X    {
X      char posstr[20];
X
X      sprintf(posstr, "\033[8;%d;%dt", ysiz, xsiz);
X      proto_out(find_port(wdes), posstr, strlen(posstr));
X    }
X    state = S_NORMAL;
X      }
X      else state = S_NORMAL;
X      break;
X    } /* end switch on state */
X
X    if (scrolled >= MAXSCROLLED)
X    {
X      if (wp->top_y != Y0) {
X        scrollup(wp, 0, ysiz, wp->top_y/inc_y);
X    wp->top_y = Y0;
X        f_y = cur_y = Y0 + (ysiz - 1) * inc_y;
X      }
X      w_redraw(wdes, FM_COPY, wp->x, wp->y, wp->w, wp->h);
X      scrolled = 0;
X    }
X  } /* end while loop for each character */
X
X  if (wp->ptr_status & LOG_BOTOM) {
X    ptr = savptr;
X    while ((ch = *ptr++) && (wp->ptr_status & LOG_BOTOM))
X      if (Cprnout(ch) == 0) wp->ptr_status = LOG_NONE;
X  }
X
X  if (wp->wi_lfd) {
X    while (ch = *savptr++)
X      putc(ch, wp->wi_lfd);
X  }
X
X  if (scrolled) {
X    if (scrolled >= MAXSCROLLED) {
X      if (wp->top_y != Y0) {
X        scrollup(wp, 0, ysiz, wp->top_y/inc_y);
X        wp->top_y = Y0;
```

```
X        cur_y = Y0 + (ysiz - 1) * inc_y;
X      }
X    }
X    w_redraw(wdes, FM_COPY, wp->x, wp->y, wp->w, wp->h);
X  }
X  else
X    if (f_mod)
X    {
X      if (!wp->insmode)
X        w_update(wdes, FM_COPY, f_x, f_y, cur_x - f_x, inc_y);
X      else
X        w_update(wdes, FM_COPY, f_x, f_y, xsiz * inc_x-f_x, inc_y);
X    }
X  wp->cur_x = cur_x;
X  wp->cur_y = cur_y;
X  wp->state = state;
X#ifdef DEBUG
X  printf ("calling w_flash\n");
X#endif
X  w_flash(wdes, 1);
X}
X
Xlineerase(wp, first, last)
Xregister struct wi_str *wp;
X{
X  register int *p;
X  long count;
X  long linespace = wp->wi_mf.wwords*wp->font->inc_y;
X
X  p = wp->wi_mf.ptr + first*linespace + Y0*wp->wi_mf.wwords - 1;
X  count = (last-first+1)*linespace;
X  while (count > 7)
X  {
X    *(((long *)p)++) = 0;
X    *(((long *)p)++) = 0;
X    *(((long *)p)++) = 0;
X    *(((long *)p)++) = 0;
X    count -= 8;
X  }
X  while (--count >= 0)
X    *++p = 0;
X}
X
Xscrollup(wp, first, nlines, amount)
Xregister struct wi_str *wp;
X{
X  register int *p1, *p2;
X  register long count;
X  int linespace = wp->wi_mf.wwords*wp->font->inc_y;
X
X  p1 = wp->wi_mf.ptr + first*linespace + Y0*wp->wi_mf.wwords;
X  p2 = p1 + linespace * amount;
X  count = (long)(nlines)*linespace;
```

```
X  while (count > 15)
X  {
X    *(((long *)p1)++) = *(((long *)p2)++);
X    *(((long *)p1)++) = *(((long *)p2)++);
X    *(((long *)p1)++) = *(((long *)p2)++);
X    *(((long *)p1)++) = *(((long *)p2)++);
X    *(((long *)p1)++) = *(((long *)p2)++);
X    *(((long *)p1)++) = *(((long *)p2)++);
X    *(((long *)p1)++) = *(((long *)p2)++);
X    *(((long *)p1)++) = *(((long *)p2)++);
X    count -= 16;
X  }
X  while (--count >= 0)
X    *(p1++) = *(p2++);
X  count = linespace * amount;
X  while (count > 7)
X  {
X    *(((long *)p1)++) = 0;
X    *(((long *)p1)++) = 0;
X    *(((long *)p1)++) = 0;
X    *(((long *)p1)++) = 0;
X    count -= 8;
X  }
X  while (--count >= 0)
X    *(p1++) = 0;
X}
X
Xscrolldn(wp, first, nlines, amount)
Xregister struct wi_str *wp;
X{
X  register int *p1, *p2;
X  register long count;
X  long linespace = wp->wi_mf.wwords*wp->font->inc_y;
X
X  p1 = wp->wi_mf.ptr + (nlines+first+amount)*linespace + Y0*wp->wi_mf.wwords;
X
X  p2 = p1 - linespace * amount;
X  count = (long)(nlines)*linespace;
X  while (count > 15)
X  {
X    *--(long *)p1 = *--(long *)p2;
X    *--(long *)p1 = *--(long *)p2;
X    *--(long *)p1 = *--(long *)p2;
X    *--(long *)p1 = *--(long *)p2;
X    *--(long *)p1 = *--(long *)p2;
X    *--(long *)p1 = *--(long *)p2;
X    *--(long *)p1 = *--(long *)p2;
X    *--(long *)p1 = *--(long *)p2;
X    count -= 16;
X  }
X  while (--count >= 0)
X    *--p1 = *--p2;
X  count = linespace * amount;
```

```
X  while (count > 7)
X  {
X    *--(long *)p1 = 0L;
X    *--(long *)p1 = 0L;
X    *--(long *)p1 = 0L;
X    *--(long *)p1 = 0L;
X    count -= 8;
X  }
X  while (--count >= 0)
X    *--p1 = 0;
X}
SHAR_EOF
chmod 0600 WINSUBR.C || echo "restore of WINSUBR.C fails"
rm -f s2_seq_.tmp
echo "You have unpacked the last part"
exit 0


From -1353073794947263855
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,ff8d193354ec2c10,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-11 16:33:05 PST
Path: utzoo!attcan!uunet!husc6!bloom-beacon!gatech!purdue!decwrl!labrea!agate!
saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i024: uuduue -- Dumas uuencode/decode programs
Keywords: shar, C
Message-ID: <6656@saturn.ucsc.edu>
Date: 12 Mar 89 00:33:05 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 817
Approved: koreth@ssyx.ucsc.edu

Submitted-by: uunet!mcvax!unido!klute (Rainer Klute)
Posting-number: Volume 2, Issue 24
Archive-name: uuduue

Here are the latest sources of Dumas' UUD/UUE package. They
were posted a while ago here in europe. They are *not*
uuencoded to give you a chance to retrieve them.

  Rainer Klute                  ----   klute@irb.informatik.uni-dortmund.de
  Universitaet Dortmund, IRB    |)||/   klute@unido.uucp, klute@unido.bitnet
  Postfach 500500               |\||\   ...uunet!mcvax!unido!klute
D-4600 Dortmund 50              ----   Tel.: +49 231 7554663


#! /bin/sh
# This is a shell archive, meaning:
# 1. Remove everything above the #! /bin/sh line.
# 2. Save the resulting text in a file.
```

```
# 3. Execute the file with /bin/sh (not csh) to create:
#    UUD.C
#    UUE.C
# This archive created: Wed Feb  8 10:57:34 1989
export PATH; PATH=/bin:/usr/bin:$PATH
if test -f 'UUD.C'
then
     echo shar: "will not over-write existing file 'UUD.C'"
else
cat << \SHAR_EOF > 'UUD.C'
/*
 * Uud -- decode a uuencoded file back to binary form.
 *
 * From the Berkeley original, modified by MSD, RDR, JPHD & WLS.
 * The Atari GEMDOS version compiled with MWC 2.x.
 * The MSDOS version with TurboC.
 * The Unix version with cc.
 * this version is made: 25 Nov 1988.
 */

/*
 * Be sure to have the proper symbol at this point. (GEMDOS, MSDOS, UNIX...)
 */

#ifndef GEMDOS
#define GEMDOS 1
#endif
/*
#ifndef UNIX
#define UNIX 1
#endif
*/
/*
#ifndef MSDOS
#define MSDOS 1
#endif
 */

#ifdef GEMDOS
#define SYSNAME "gemdos"
#define SMALL 1
#endif
#ifdef MSDOS
#define SYSNAME "msdos"
#define SMALL 1
#endif
#ifdef UNIX
#define SYSNAME "unix"
#endif

#include <stdio.h>

#ifdef GEMDOS
```

```
#include <osbind.h>
#define Error(n)  { Bconin(2); exit(n); }
#define WRITE   "wb"
#else
#define Error(n)  exit(n)
#define WRITE   "w"
#endif

#define loop  while (1)

extern FILE *fopen();
extern char *strcpy();
extern char *strcat();

char *getnword();

#define MAXCHAR 256
#define LINELEN 256
#define FILELEN 64
#define NORMLEN 60 /* allows for 80 encoded chars per line */

#define SEQMAX 'z'
#define SEQMIN 'a'
char seqc;
int first, secnd, check, numl;

FILE *in, *out;
char *pos;
char ifname[FILELEN], ofname[FILELEN];
char *source = NULL, *target = NULL;
char blank, part = '\0';
int partn, lens;
int debug = 0, nochk = 0, onedone = 0;
int chtbl[MAXCHAR], cdlen[NORMLEN + 3];

main(argc, argv) int argc; char *argv[];
{
     int mode;
     register int i, j;
     char *curarg;
     char dest[FILELEN], buf[LINELEN];

     if (argc < 2) {
          format("Almost foolproof uudecode v3.4 (%s) 25-Nov-88\n",
               SYSNAME);
          format("\n");
          format("Usage: uud [-n] [-d] [-s dir] [-t dir] input-file\n");
          format("\n");
          format("Option: -n -> No line sequence check\n");
          format("Option: -d -> Debug/verbose mode\n");
          format("Option: -s + Source directory for all input files\n");
          format("  (MUST be terminated by directory separator)\n");
          format("Option: -t + Target directory for all output files\n");
```

```
     format("  (MUST be terminated by directory separator)\n");
     format("If input-file is - then stdin is used as input-file\n");
     Error(1);
}

curarg = argv[1];

while (curarg[0] == '-') {
     if (((curarg[1] == 'd') || (curarg[1] == 'D')) &&
         (curarg[2] == '\0')) {
          debug = 1;
     } else if (((curarg[1] == 'n') || (curarg[1] == 'N')) &&
             (curarg[2] == '\0')) {
          nochk = 1;
     } else if (((curarg[1] == 't') || (curarg[1] == 'T')) &&
             (curarg[2] == '\0')) {
          argv++;
          argc--;
          if (argc < 2) {
               format("uud: Missing target directory.\n");
               Error(15);
          }
          target = argv[1];
          if (debug)
               format("Target dir = %s\n",target);
     } else if (((curarg[1] == 's') || (curarg[1] == 'S')) &&
             (curarg[2] == '\0')) {
          argv++;
          argc--;
          if (argc < 2) {
               format("uud: Missing source directory.\n");
               Error(15);
          }
          source = argv[1];
          if (debug)
               format("Source dir = %s\n",source);
     } else if (curarg[1] != '\0') {
          format("uud: Unknown option <%s>\n", curarg);
          Error(15);
     } else
          break;
     argv++;
     argc--;
     if (argc < 2) {
          format("uud: Missing file name.\n");
          Error(15);
     }
     curarg = argv[1];
}

if ((curarg[0] == '-') && (curarg[1] == '\0')) {
     in = stdin;
     strcpy(ifname, "<stdin>");
```

```
    } else {
         if (source != NULL) {
              strcpy(ifname, source);
              strcat(ifname, curarg);
         } else
              strcpy(ifname, curarg);
         if ((in = fopen(ifname, "r")) == NULL) {
              format("uud: Can't open %s\n", ifname);
              Error(2);
         }
         numl = 0;
    }

/*
 * Set up the default translation table.
 */
    for (i = 0; i < ' '; i++) chtbl[i] = -1;
    for (i = ' ', j = 0; i < ' ' + 64; i++, j++) chtbl[i] = j;
    for (i = ' ' + 64; i < MAXCHAR; i++) chtbl[i] = -1;
    chtbl['`'] = chtbl[' '];    /* common mutation */
    chtbl['~'] = chtbl['^'];    /* an other common mutation */
    blank = ' ';
/*
 * set up the line length table, to avoid computing lotsa * and / ...
 */
    cdlen[0] = 1;
    for (i = 1, j = 5; i <= NORMLEN; i += 3, j += 4)
         cdlen[i] = (cdlen[i + 1] = (cdlen[i + 2] = j));
/*
 * search for header or translation table line.
 */
    loop {   /* master loop for multiple decodes in one file */
         partn = 'a';
         loop {
              if (fgets(buf, sizeof buf, in) == NULL) {
                   if (onedone) {
                        if (debug) format("End of file.\n");
                        exit(0);
                   } else {
                        format("uud: No begin line.\n");
                        Error(3);
                   }
              }
              numl++;
              if (strncmp(buf, "table", 5) == 0) {
                   gettable();
                   continue;
              }
              if (strncmp(buf, "begin", 5) == 0) {
                   break;
              }
         }
         lens = strlen(buf);
```

```
          if (lens) buf[--lens] = '\0';
#ifdef SMALL
          if ((pos = getnword(buf, 3))) {
               strcpy(dest, pos);
          } else
#else
          if(sscanf(buf,"begin%o%s", &mode, dest) != 2)
#endif
          {
               format("uud: Missing filename in begin line.\n");
               Error(10);
          }

          if (target != NULL) {
               strcpy(ofname, target);
               strcat(ofname, dest);
          } else
               strcpy(ofname, dest);

          if((out = fopen(ofname, WRITE)) == NULL) {
               format("uud: Cannot open output file: %s\n", ofname);
               Error(4);
          }
          if (debug) format("Begin uudecoding: %s\n", ofname);
          seqc = SEQMAX;
          check = nochk ? 0 : 1;
          first = 1;
          secnd = 0;
          decode();
          fclose(out);
#ifdef UNIX
          chmod(ofname, mode);
#endif
          onedone = 1;
          if (debug) format("End uudecoding: %s\n", ofname);
     }    /* master loop for multiple decodes in one file */
}

/*
 * Bring back a pointer to the start of the nth word.
 */
char *getnword(str, n) register char *str; register int n;
{
     while((*str == '\t') || (*str == ' ')) str++;
     if (! *str) return NULL;
     while(--n) {
          while ((*str != '\t') && (*str != ' ') && (*str)) str++;
          if (! *str) return NULL;
          while((*str == '\t') || (*str == ' ')) str++;
          if (! *str) return NULL;
     }
     return str;
}
```

```
/*
 * Install the table in memory for later use.
 */
gettable()
{
     char buf[LINELEN];
     register int c, n = 0;
     register char *cpt;

     for (c = 0; c <= MAXCHAR; c++) chtbl[c] = -1;

again:    if (fgets(buf, sizeof buf, in) == NULL) {
          format("uud: EOF while in translation table.\n");
          Error(5);
     }
     numl++;
     if (strncmp(buf, "begin", 5) == 0) {
          format("uud: Incomplete translation table.\n");
          Error(6);
     }
     cpt = buf + strlen(buf) - 1;
     *cpt = ' ';
     while (*(cpt) == ' ') {
          *cpt = 0;
          cpt--;
     }
     cpt = buf;
     while (c = *cpt) {
          if (chtbl[c] != -1) {
               format("uud: Duplicate char in translation table.\n");
               Error(7);
          }
          if (n == 0) blank = c;
          chtbl[c] = n++;
          if (n >= 64) return;
          cpt++;
     }
     goto again;
}

/*
 * copy from in to out, decoding as you go along.
 */

decode()
{
     char buf[LINELEN], outl[LINELEN];
     register char *bp, *ut;
     register int *trtbl = chtbl;
     register int n, c, rlen;
     register unsigned int len;
```

```
    loop {
         if (fgets(buf, sizeof buf, in) == NULL) {
              format("uud: EOF before end.\n");
              fclose(out);
              Error(8);
         }
         numl++;
         len = strlen(buf);
         if (len) buf[--len] = '\0';
/*
 * Is it an unprotected empty line before the end line ?
 */
         if (len == 0) continue;
/*
 * Get the binary line length.
 */
         n = trtbl[*buf];
         if (n >= 0) goto decod;
/*
 * end of uuencoded file ?
 */
         if (strncmp(buf, "end", 3) == 0) return;
/*
 * end of current file ? : get next one.
 */
         if (strncmp(buf, "include", 7) == 0) {
              getfile(buf);
              continue;
         }
         format("uud: Bad prefix line %d in file: %s\n",numl, ifname);
         if (debug) format("Bad line =%s\n",buf);
         Error(11);
/*
 * Sequence checking ?
 */
decod:        rlen = cdlen[n];
/*
 * Is it the empty line before the end line ?
 */
         if (n == 0) continue;
/*
 * Pad with blanks.
 */
         for (bp = &buf[c = len];
              c < rlen; c++, bp++) *bp = blank;
/*
 * Verify if asked for.
 */
         if (debug) {
              for (len = 0, bp = buf; len < rlen; len++) {
                   if (trtbl[*bp] < 0) {
                        format(
    "Non uuencoded char <%c>, line %d in file: %s\n", *bp, numl, ifname);
```

```
                    format("Bad line =%s\n",buf);
                    Error(16);
               }
               bp++;
          }
     }
/*
 * All this just to check for uuencodes that append a 'z' to each line....
 */
     if (secnd && check) {
          secnd = 0;
          if (buf[rlen] == SEQMAX) {
               check = 0;
               if (debug) format("Sequence check turned off (2).\n");
          } else
               if (debug) format("Sequence check on (2).\n");
     } else if (first && check) {
          first = 0;
          secnd = 1;
          if (buf[rlen] != SEQMAX) {
               check = 0;
               if (debug) format("No sequence check (1).\n");
          } else
               if (debug) format("Sequence check on (1).\n");
     }
/*
 * There we check.
 */
     if (check) {
          if (buf[rlen] != seqc) {
               format("uud: Wrong sequence line %d in %s\n",
                    numl, ifname);
               if (debug)
                    format(
    "Sequence char is <%c> instead of <%c>.\n", buf[rlen], seqc);
               Error(18);
          }
          seqc--;
          if (seqc < SEQMIN) seqc = SEQMAX;
     }
/*
 * output a group of 3 bytes (4 input characters).
 * the input chars are pointed to by p, they are to
 * be output to file f.n is used to tell us not to
 * output all of them at the end of the file.
 */
     ut = outl;
     len = n;
     bp = &buf[1];
     while (n > 0) {
          *(ut++) = trtbl[*bp] << 2 | trtbl[bp[1]] >> 4;
          n--;
          if (n) {
```

```
                    *(ut++) = (trtbl[bp[1]] << 4) |
                           (trtbl[bp[2]] >> 2);
                    n--;
               }
               if (n) {
                    *(ut++) = trtbl[bp[2]] << 6 | trtbl[bp[3]];
                    n--;
               }
               bp += 4;
          }
          if ((n = fwrite(outl, 1, len, out)) <= 0) {
               format("uud: Error on writing decoded file.\n");
               Error(18);
          }
     }
}

/*
 * Find the next needed file, if existing, otherwise try further
 * on next file.
 */
getfile(buf) register char *buf;
{
     if ((pos = getnword(buf, 2)) == NULL) {
          format("uud: Missing include file name.\n");
          Error(17);
     } else
          if (source != NULL) {
               strcpy(ifname, source);
               strcat(ifname, pos);
          } else
               strcpy(ifname, pos);
#ifdef GEMDOS
     if (Fattrib(ifname, 0, 0) < 0)
#else
     if (access(ifname, 04))
#endif
     {
          if (debug) {
               format("Cant find: %s\n", ifname);
               format("Continuing to read same file.\n");
          }
     }
     else {
          if (freopen(ifname, "r", in) == in) {
               numl = 0;
               if (debug)
                    format("Reading next section from: %s\n", ifname);
          } else {
               format("uud: Freopen abort: %s\n", ifname);
               Error(9);
          }
     }
```

```
     loop {
          if (fgets(buf, LINELEN, in) == NULL) {
               format("uud: No begin line after include: %s\n", ifname);
               Error(12);
          }
          numl++;
          if (strncmp(buf, "table", 5) == 0) {
               gettable();
               continue;
          }
          if (strncmp(buf, "begin", 5) == 0) break;
     }
     lens = strlen(buf);
     if (lens) buf[--lens] = '\0';
/*
 * Check the part suffix.
 */
     if ((pos = getnword(buf, 3)) == NULL ) {
          format("uud: Missing part name, in included file: %s\n", ifname);
          Error(13);
     } else {
          part = *pos;
          partn++;
          if (partn > 'z') partn = 'a';
          if (part != partn) {
               format("uud: Part suffix mismatch: <%c> instead of <%c>.\n",
                    part, partn);
               Error(14);
          }
          if (debug) format("Reading part %c\n", *pos);
     }
}

/*
 * Printf style formatting. (Borrowed from MicroEmacs by Dave Conroy.)
 * A lot smaller than the full fledged printf.
 */
/* VARARGS1 */
format(fp, args) char *fp;
{
     doprnt(fp, (char *)&args);
}

doprnt(fp, ap)
register char *fp;
register char *ap;
{
     register int  c, k;
     register char *s;

     while ((c = *fp++) != '\0') {
          if (c != '%')
               outc(c);
```

```
          else {
               c = *fp++;
               switch (c) {
               case 'd':
                    puti(*(int *)ap, 10);
                    ap += sizeof(int);
                    break;

               case 's':
                    s = *(char **)ap;
                    while ((k = *s++) != '\0')
                         outc(k);
                    ap += sizeof(char *);
                    break;

               case 'c':
                    outc(*(int *)ap);
                    ap += sizeof(int);
                    break;

               default:
                    outc(c);
               }
          }
     }
}

/*
 * Put integer, in radix "r".
 */
puti(i, r)
register unsigned int  i;
register unsigned int  r;
{
     register unsigned int  q, s;

     if ((q = i / r) != 0)
          puti(q, r);
     s = i % r;
     if (s <= 9)
          outc(s + '0');
     else
          outc(s - 10 + 'A');
}
outc(c) register char c;
{
#ifdef GEMDOS
     if (c == '\n') Bconout(2, '\r');
     Bconout(2, c);
#else
     putchar(c);
#endif
}
```

```
SHAR_EOF
fi
if test -f 'UUE.C'
then
     echo shar: "will not over-write existing file 'UUE.C'"
else
cat << \SHAR_EOF > 'UUE.C'
/*
 *
 * Uue -- encode a file so that it's printable ascii, short lines
 *
 * Slightly modified from a version posted to net.sources a while back,
 * and suitable for compilation on the IBM PC
 *
 * modified for Lattice C on the ST - 11.05.85 by MSD
 * modified for ALCYON on the ST -    10-24-86 by RDR
 * modified a little more for MWC...  02/09/87 by JPHD
 * (An optional first argument of the form: -nnumber (e.g. -500), will
 * produce a serie of files that long, linked by the include statement,
 * such files are automatically uudecoded by the companion program.)
 * More mods, - ...            05/06/87 by jphd
 * Mods for TOPS 20, and more.     08/06/87 by jphd
 *     (remove freopen and rindex...change filename generation...)
 * (A lot more to do about I/O speed, avoiding completely the stdio.h...)
 *
 */


#include <stdio.h>
#include <ctype.h>

#define USAGE

/* ENC is the basic 1 character encoding function to make a char printing */
#define ENC(c) (((c) & 077) + ' ')

extern FILE  *fopen();
FILE *fp, *outp;
char ofname[80];
int lenofname;
int stdo = 0;

#ifdef ST
#define READ "rb"
#else
#define READ "r"
#endif

int part = 'a', chap = 'a';
#define SEQMAX 'z'
#define SEQMIN 'a'
char seqc = SEQMAX;
```

```
int split = 0; fileln = 32000;

main(argc, argv)
int argc; char *argv[];
{
     char *fname;

     if (argc < 2) {
          fprintf(stderr, "Almost foolproof uuencode v3.1 06 Aug 1987\n");
          fprintf(stderr, "Usage: uue [-n] inputfile [-]\n");
          exit(2);
          }
     if (argv[1][0] == '-') {
          fileln = -atoi(argv[1]);
          if (fileln <= 0) {
               fprintf(stderr, "Wrong file length arg.\n");
               exit();
          }
          split = 1;
          argv++;
          argc--;
     }
     if ((fp=fopen(argv[1], READ))==NULL) {  /* binary input !!! */
          fprintf(stderr,"Cannot open %s\n",argv[1]);
          exit(1);
     }
     strcpy(ofname, argv[1]);
     fname = ofname;
     do {
          if (*fname == '.')
               *fname = '\0';
     } while (*fname++);
          /* 8 char prefix + .uue -> 12 chars MAX */
     lenofname = strlen(ofname);
     if (lenofname > 8) ofname[8] = '\0';
     strcat(ofname,".uue");
     lenofname = strlen(ofname);
     if (!split && (argc > 2) && (argv[2][0] == '-')) {
          stdo = 1;
          outp = stdout;
      } else {
           makename();
           if((outp = fopen(ofname, "w")) == NULL) {
                fprintf(stderr,"Cannot open %s\n", ofname);
                exit(1);
                }
     }
     maketable();
     fprintf(outp,"begin %o %s\n", 0644, argv[1]);
     encode();
     fprintf(outp,"end\n");
     fclose(outp);
     exit(0);
```

```
}

/* create ASCII table so a mailer can screw it up and the decode
 * program can restore the error.
 */
maketable()
{
     register int i, j;

     fputs("table\n", outp);
     for(i = ' ', j = 0; i < '`' ; j++) {
          if (j == 32)
               putc('\n', outp);
          fputc(i++, outp);
     }
     putc('\n', outp);
}

/*
 * Generate the names needed for single and multiple part encoding.
 */
makename()
{
     if (split) {
          ofname[lenofname - 1] = part;
          ofname[lenofname - 2] = chap;
     }
}

/*
 * copy from in to out, encoding as you go along.
 */
encode()
{
     char buf[80];
     register int i, n;
     register int lines;
     lines = 6;

     for (;;) {
          n = fr(buf, 45);
          putc(ENC(n), outp);
          for (i = 0; i < n; i += 3)
                outdec(&buf[i]);
          putc(seqc, outp);
          seqc--;
          if (seqc < SEQMIN) seqc = SEQMAX;
          putc('\n', outp);
          ++lines;
          if (split && (lines > fileln)) {
               part++;
               if (part > 'z') {
                    part = 'a';
```

```
                    if (chap == 'z')
                         chap = 'a'; /* loop ... */
                    else
                         chap++;
               }
               makename();
               fprintf(outp,"include %s\n",ofname);
               fclose(outp);
               if((outp = fopen(ofname, "w")) == NULL) {
                    fprintf(stderr,"Cannot open %s\n",ofname);
                    exit(1);
               }
               maketable();
               fprintf(outp,"begin part %c %s\n",part,ofname);
               lines = 6;
          }
          if (n <= 0)
               break;
     }
}

/*
 * output one group of 3 bytes, pointed at by p, on file f.
 */
outdec(p)
register char *p;
{
     register int c1, c2, c3, c4;

     c1 = *p >> 2;
     c2 = (*p << 4) & 060 | (p[1] >> 4) & 017;
     c3 = (p[1] << 2) & 074 | (p[2] >> 6) & 03;
     c4 = p[2] & 077;
     putc(ENC(c1), outp);
     putc(ENC(c2), outp);
     putc(ENC(c3), outp);
     putc(ENC(c4), outp);
}

/* fr: like read but stdio */
int fr(buf, cnt)
register char *buf;
register int cnt;
{
     register int c, i;
     for (i = 0; i < cnt; i++) {
          c = fgetc(fp);
          if (feof(fp))
               return(i);
          buf[i] = c;
     }
     return (cnt);
}
```

```
SHAR_EOF
fi
exit 0
#    End of shell archive
```

```
From 6242739620240585907
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,ce072681e72a7f43,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-11 16:36:14 PST
Path: utzoo!attcan!uunet!husc6!bloom-beacon!gatech!purdue!decwrl!labrea!agate!
saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i025: mwstring -- dpANS string.h for MWC
Keywords: MWC
Message-ID: <6659@saturn.ucsc.edu>
Date: 12 Mar 89 00:36:14 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 147
Approved: koreth@ssyx.ucsc.edu

Submitted-by: csrobe@cs.wm.edu (Chip Roberson)
Posting-number: Volume 2, Issue 25
Archive-name: mwstring

Here is a reasonable implementation of the Draft Proposed American National
Standard (dpANS) header file for strings in C.  It should be reasonably
consistent with the dpANS while representing MWC's implementation.

Enjoy,
-c
|Chip Roberson                ARPANET:  csrobe@cs.wm.edu                  |
|Dept of Comp. Sci.                     csrobe@icase.edu                  |
|College of William and Mary  BITNET:   #csrobe@wmmvs.bitnet              |
|Williamsburg, VA 23185       UUCP:     ...!uunet!pyrdc!gmu90x!wmcs!csrobe|

PS: Anyone wishing to join a mailing list to discuss the timely topic of
animal-rights, can send their name and e-mail address to csrobe@cs.wm.edu,
and I'll add you name to the list.

     "It takes 40 dumb animals to make a fur coat,
         and just one dumb animal to wear it."      -European TV commercial

---------------snip---------------snip---------------snip---------------
/* [csr]                          10 Mar 1988
 *   string.h -- declarations for handling strings
 *
 *
 *   This is a replacement string.h file for MWC.
 *   It was written using _Notes_on_the_Draft_C_Standard_
 *   written by Thomas Plum, the MWC "Lexicon" for release 2.0,
```

```
 *   and MWC _Release_Notes_ for Update Version 3.0.6.
 *
 *   I guess I could claim a copyright, but why?  Please feel free
 *   to share this with others and correct where necessary.
 *
 *   C. Roberson   csrobe@cs.wm.edu
 */

#ifndef _STRING_HDR_
#define _STRING_HDR_


/*
 *   void *   memchr(const void *s, int c, size_t n);
 */
char *   memchr( /* char *region, unsigned int character, unsigned int n */ );

/*
 *   int  memcmp(const void *s1, const void *s2, size_t n);
 */
int  memcmp( /* char *region1, char *region2, unsigned int count */ );

/*
 *   void *   memcpy(const void *s1, const void *s2, size_t n);
 */
char *   memcpy( /* char *region1, char *region2, unsigned int n */ );

/*
 *   void *   memmove(const void *s1, const void *s2, size_t n);
 */

/*
 *   void *   memset(const void *s, int c, size_t n);
 */
char *   memset( /* char *buffer, int character, unsigned int n */ );

/*
 *   char *   strcat(char *s1, const char *s2);
 */
char *   strcat( /* char *string1, char *string2 */ );

/*
 *   char *   strchr(const char *s, int c);
 */
char *   strchr( /* char * string, int character */ );

/*
 *   int  strcmp(const char *s1, const char *s2);
 */
int  strcmp( /* char *string1, char *string2 */ );

/*
 *   int  strcoll(const char *s1, const char *s2);
```

```
 */

/*
 *   int  strcpy(const char *s1, const char *s2);
 */
char *   strcpy( /* char *string1, char *string2 */ );

/*
 *   size_t   strcspn(const char *s1, const char *s2);
 */
unsigned int strcspn( /* char *string1, char *string2 */ );

/*
 *   char *   strerror(int errnum);
 */
char *   strerror( /* int error */ );

/*
 *   size_t   strlen(const char *s);
 */
int  strlen( /* char * string */ );

char *   strncat( /* char *string1, char *string2, unsigned n */ );

/*
 *   char *   strncmp(const char *s1, const char *s2, size_t n);
 */
int  strncmp( /* char *string1, char *string2, unsigned n */ );

/*
 *   char *   strncpy(char *s1, const char *s2, size_t n);
 */
char *   strncpy( /* char *string1, char *string2, unsigned n */ );

/*
 *   char *   strpbrk(const char *s1, const char *s2);
 */
char *   strpbrk( /* char *string1, char *string2 */ );

/*
 *   char *   strrchr(const char *s1, int c);
 */
char *   strrchr( /* char *string1, int character */ );

/*
 *   size_t   strspn(const char *s1, const char *s2);
 */
unsigned int  strspn( /* char *string1, char *string2 */ );

/*
 *   char *   strstr(const char *s1, const char *s2);
 */
char *   strstr( /* char *string1, char *string2 */ );
```

```
/*
 *   char *   strtok(char *s1, char *s2);
 */
char *   strtok( /* char *string1, char *string2 */ );

/*
 *   size_t   strxfrm(char *to, const char *from, size_t maxsize);
 */

#endif


From 2635831702499857304
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,167d5f3bc9f6c134,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-11 16:42:25 PST
Path: utzoo!attcan!uunet!husc6!ogccse!blake!uw-beaver!cornell!batcomputer!
sun.soe.clarkson.edu!rpi!leah!csd4.milw.wisc.edu!bionet!agate!saturn!
ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i026: mailtruk -- GFA BASIC uucp software
Keywords: shar, GFA
Message-ID: <6660@saturn.ucsc.edu>
Date: 12 Mar 89 00:42:25 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1522
Approved: koreth@ssyx.ucsc.edu

Submitted-by: john@logajan.mn.org (John Logajan)
Posting-number: Volume 2, Issue 26
Archive-name: mailtruk

Below are five files generically named MailTruk Release V1.0.
MailTruk is a GFA Basic (2.0) implementation of uucp/uucico.
MailTruk is intended to allow non-unix/non-C Atari ST's email
access to the Usenet community.

#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      mailtruk.doc
#      mailtruk.lst
#      protocol.doc
#      readmail.lst
#      writmail.lst
#
sed 's/^X//' << 'SHAR_EOF' > mailtruk.doc &&
X
X                       MailTruk Documentation
```

X   by
X   John Logajan, January 1989
X   4248 Hamline Ave
X   Arden Hills, MN 55112
X   john@logajan.mn.org
X
X
XINTRODUCTION TO THE NETWORK
X
XWelcome to the world of uucp style data exchange. Uucp stands for
XUnix to Unix copy. Unix is an operating system used by a large
Xnumber of computers. A large network of computers running uucp
Xcompatible data exchange has evolved over the last several years.
XThis network encompasses university, corporate, governmental, and
Xindividually owned computers. It is estimated that there are over
X10,000 computers on the net, with approximately 400,000 users.
XThe two primary functions of the net are private electronic mail
Xand "broadly cast" electronic news transfer. The Usenet News
Xconsists of some 350 subject categories into which several
Xthousand articles a day are posted by users. An average day's
Xnews is typically four million characters -- and growing. Anyone
Xreceiving a full newsfeed is more and more feeling the need for
Xfaster data communications links, since such volume at 2400 baud
Xrequires five hours if uncompressed or 2.5 hours if compressed.
X
X
XINTRODUCTION TO THE MAILTRUK, READMAIL, AND WRITMAIL PROGRAMS
X
XMailTruk, ReadMail, and WritMail are uucp compatible
Xcommunications programs that run on Atari ST computers.   The
Xprograms are written in GFA Basic (2.0) and are freely distributed
Xas a readable listings in the public domain. Files included in
Xthe distribution include:
XMAILTRUK.LST, READMAIL.LST, WRITMAIL.LST, MAILTRUK.DOC,
XPROTOCOL.DOC
X
XWritMail is used to write your outgoing mail. Writmail's main
Xtask is to format your message so that other computers on the net
Xrecognize what you have sent and where it is going. Each, in
Xturn, passes it along until it hopefully reaches your intended
Xrecipient.
X
XMailTruk is the actual communications program. It first dials
Xyour neighbor site, establishes protocol agreement, and then sends
Xthe messages, if any, that you have previously prepared with
XWritMail. When MailTruk has finished sending, it requests to
Xreceive any mail that the remote site has pending for you. After
Xit receives all incoming mail, MailTruk terminates the connection.
X
XReadMail allows you to read your mail in a convenient manner. The
Xmain objection to reading you mail files directly with GEM
Xcommands is that Unix machines use linefeed characters as end-of-
Xline indicators rather than the carriage return character used by

XGEM. Thus, direct reading of mail files results in a confusing
Xpresentation.
X
XAlthough MailTruk can transfer the file types used to send the
XNetwork News, additional file pre- and post-processing is required
Xto read or write News articles. ReadMail and WritMail DO NOT
Xhandle the News format. Uncompressed News would be a fairly
Xstraight forward task to handle, but compressed and batched News
Xwill require additional sophistication. Let me know if you feel
Xmotivated to write such a program!
X
X
XSETTING EVERYTHING UP
X
XFirst -- You need a site to call, account information for that
Xsite, and a modem. Since this program is being distributed over
Xthe net, if you track back from where this program came from,
Xyou're bound to run into somebody who knows your local net
Xarrangement. (I pay $25 a year to a guy who runs a "domain park"
Xfor my access. He, in turn, calls other bigger sites -- and so
Xon.)
X
XSecond -- All mail going to a particular site is placed in its own
Xdirectory. All mail coming to your site is placed in its own
Xdirectory. You should create these directories before running the
Xprograms. I am site -logajan-, I call one site -bungia-. So, I
Xcreated a directory called MAIL on my C: drive where I put all my
Xprograms, and I created two sub-directories, BUNGIA and LOGAJAN,
Xi.e. C:\MAIL\BUNGIA\
X   C:\MAIL\LOGAJAN\
X
XThird -- All three programs are totally self-contained in that
Xthey do not share an external common configuration file. So you
Xwill have to insure that each of the important variables,
Xincluding the directory names mentioned above, in each of the
Xprograms is correct and consistent. The variables are grouped
Xnear the beginning of each program. You WILL have to customize
Xthese variables for these programs to work!
X
XFourth -- If you have multiple sites to call, you will have to
Xmake multiple versions of the MailTruk program (or modify it to
Xhandle multiple cases.) You will also want to write a program
Xthat recognizes messages which are just passing through, and
X"rewrites" the pertinent addresses, and places them back out in
Xthe outbound message area.
X
XFifth -- Otherwise you are a "leaf" site (you only call one
Xmachine) and any mail with your machine name in it is assumed to
Xbe to you, no matter what user name or additional path is in the
XTo: address line.
X
XSixth -- Paths; the From: and To: portion of messages; are a
Xsomewhat mysterious subject. Basically you have to know, at a

```
Xminimum, the recipient's user-name and machine-name, and the path
Xto the nearest smart mailer site.   Otherwise you can specify the
Xentire path -- but some smart sites end up sending it the route
Xthey think is best anyhow.
X
X  If I want to send to jack at atari, I enter the path as:
X     bungia!atari!jack     Note:  nearest!next!next!farthest!user
X     (my neighbor, bungia, knows the tortuous route to atari)
X
X  or bungia!umn-cs!rutgers!att.att.com!wally if I want to get to
X     wally at att and know the complete path.
X
XYou will often see @'s and %'s in paths.  Basically you just read
Xthese in the other direction, i.e.:
X
X     john@logajan.mn.org@bungia@shamash@nic.mr.net   is an actual
X     path TO my machine from nic.mr.net.
X
XHowever, note that WritMail forces you to state the path with at
Xleast one (!) symbol.  Your neighbor's machine is always the first
Xname in the path, therefore you could do:
X
X     bungia!ben@franklin.electric.com
X
X
XSeventh -- Hopefully there is enough information in the listings
Xfor you to set up your own site and get it operational.
X                           Good luck.
X
SHAR_EOF
chmod 0600 mailtruk.doc || echo "restore of mailtruk.doc fails"
sed 's/^X//' << 'SHAR_EOF' > mailtruk.lst &&
XRem  MAILTRUK V1.0 -- GFA Basic 2.0 / Atari ST implementation of uucp/uucico.
XRem                       --- PUBLIC DOMAIN ---
XRem  This program dials up computer sites running Unix or Unix clone uucp,
XRem      allowing the transfer of e-mail, news and other data files.
XRem
XRem       Totally original code by:  John Logajan, January 1989.
XRem                                  4248 Hamline Ave
XRem                                  Arden Hills, MN 55112
XRem      (john@logajan.mn.org  or  ...rutgers!bungia!logajan!john)
XRem
XRem    The next eight lines must be customized for each user/site.
XRem
XMysite$="logajan"              ! Your site machine name.  (You choose!)
XMyuser$="logajan"              ! Your user/account name for remote site.
XMypw$="youguess"               ! Your password for remote site.
XSitedir$="\mail\bungia"        ! Directory of files to send only to that site.
XSitercv$="\mail\"+Mysite$      ! Directory for all incoming files.
XSiteph$="123-4567"             ! Remote site telephone number.
XRetry!=True                    ! Should we retry dialing if busy/no answer.
XVoid Xbios(15,4,0,&H88,1,1,-1) ! RS232 configuration/baud rate. (See below.)
XRem
```

```
XRem Configure RS232 - Xbios(15,baud,flow,ctrl,rst,xst,scr).
XRem baud = 0/19200, 1/9600, 4/2400, 7/1200, 9/300.
XRem flow = 0/None. (UUCP forbids xoff/xon! ST's RTS/CTS works goofy!)
XRem ctrl = 8bits, 1stop, noparity.
XRem rst = recv enabled.
XRem xst = xmit enabled.
XRem scr = not used.
XRem
XRwinsiz%=3           ! Number of packets in receive window. (Standard=3)
XRpktsiz%=64          ! Number of bytes in a receive packet. (Standard=64)
XRem
XRem Assembler code for calculating the checksum (speedy.)  Does the check-
XRem sum calculation on a packetsize worth of data starting at Buffer.
XRem The control byte is also included in the sum and everthing is sub-
XRem tracted from AAAA hex.
XRem            Checksum=C:Chk%(L:buffer,W:packetsize,W:controlbyte)
XRem
XData "206F0004322F0008342F000A203C0000FFFF4243E358280042851A18D0453A00B345"
XData "D645B0846E02B7400441000166E40242FFFFB142303CAAAA904202800000FFFF4E75"
XRead A$
XRead B$
XA$=A$+B$
XChk$=""
XFor J%=1 To Len(A$) Step 2
X  Chk$=Chk$+Chr$(Val("&H"+Mid$(A$,J%,2)))
XNext J%
XChk%=Varptr(Chk$)
XRem
XRem Assembler code for RS232 block input (speedy.)  Puts x chars into a
XRem buffer, or times out after 12 seconds.
XRem            Remainder=C:Rcv%(L:buffer,W:byteswanted)
XRem
XData "286F0004382F000842A73F3C00204E415C8F2E002A39000004BA068500000960"
XData "3F3C00013F3C00014E4D588F4A4067143F3C00013F3C00024E4D588F18C05344"
XData "670C60DC2C39000004BA9C856DD22F073F3C00204E415C8F20044E75"
XRead A$
XRead B$
XA$=A$+B$
XRead B$
XA$=A$+B$
XRcv$=""
XFor J%=1 To Len(A$) Step 2
X  Rcv$=Rcv$+Chr$(Val("&H"+Mid$(A$,J%,2)))
XNext J%
XRcv%=Varptr(Rcv$)
XDim Rcxx%(Rpktsiz%/4+2)            ! Build an internal input buffer.
XRcvbf%=Varptr(Rcxx%(0))
XDim Wfil$(2)
XWfm=0
XWfp%=Varptr(Wfm)                   ! Build a frame buffer.
XRs232p%=Xbios(14,0)                ! Get RS232 TOS buffer info pointer.
XRinsiz%=((Rpktsiz%+6)*Rwinsiz%+128)
XDim Rin%(Rinsiz%/4+1)              ! Make a large RS232 TOS input buffer.
```

```
XLpoke Rs232p%,Varptr(Rin%(0))
XDpoke Rs232p%+4,Rinsiz%
XDpoke Rs232p%+6,0                    ! Flag it empty.
XDpoke Rs232p%+8,0
XOpen "",#2,"AUX:"                    ! Open RS232 port as file #2.
XRem
XRem   You will probably need to customize the dialing part of the program,
XRem   depending upon the quirks of the site you are trying to call, and
XRem   your modem response codes.
XRem
XPrint "Dialing.    Push SPACE key to abort connection attempt."
XRepeat
X  @Ostr("ATDT"+Siteph$+"\0D")         !  Dial remote site.
X  @Waitfor("CONNECT",60)
X  Exit If Not Retry!
X  If Not Found!
X    Print "Busy/no answer. Dialing again."
X  Endif
XUntil Found!
XPrint "Connected."
XRetry!=False
XPause 200
XOut 1,&HD                            ! Send a couple of CR's to prime remote.
XPause 50
XOut 1,&HD
XPause 50
XOut 1,&HD
XPause 50
XOut 1,&HD
X@Waitfor("ogin:",30)
XPause 50
X@Ostr(Myuser$+"\0D")
X@Waitfor("sword:",30)
X@Ostr(Mypw$+"\0D")
X@Waitfor("Shere",30)
XPrint "Logged in."
XRem
XRem Initialization sequence.
XRem
X@Ostr("\10S"+Mysite$+"\0A")
X@Waitfor("ROK",30)
X@Waitfor("P",30)
X@Waitfor("g",5)
X@Ostr("\10Ug\0A")
XInita!=False
XInitb!=False
XInitc!=False
XFor Wf%=1 To 10                 ! INIT with window size / data segement size.
X  If Inita!=False Or Initb!=False
X    Wcc%=&H38 Or Rwinsiz%
X    @Wctlpkt(2)                      ! inita window size.
X  Endif
X  @Waitframe
```

```
X  If Found! And (Cntrol% And &HF8)=&H38
X    Inita!=True
X    Xwinsiz%=Cntrol% And 7
X    Rpktcod%=Int(Log(Rpktsiz%)/Log(2)-4.9)
X    Wcc%=&H30 Or Rpktcod%
X    @Wctlpkt(2)                      ! initb packet size.
X    @Waitframe
X  Endif
X  If Found! And (Cntrol% And &HF8)=&H30
X    Initb!=True
X    Xpktcod%=(Cntrol% And 7)+1
X    Xpktsiz%=32*2^(Xpktcod%-1)
X    Xpktsizh%=Xpktsiz%+6
X  Endif
X  Exit If Inita! And Initb!
XNext Wf%
XIf Initb!
X  Wcc%=&H28 Or Rwinsiz%
X  @Wctlpkt(2)                       ! initc window size.
X  For Wf%=1 To 3
X    @Waitframe
X    If Found! And (Cntrol% And &HF8)=&H28
X      Initc!=True
X    Endif
X    Exit If Initc!
X  Next Wf%
XEndif
XIf Inita!=False Or Initb!=False Or Initc!=False
X  Print "Failed initilization."
X  @Hangup
XEndif
XPrint "They want a sending window size = ";Xwinsiz%
XPrint "They want a sending data packet size = ";Xpktsiz%
XDim Xbuf%(2*Xpktsizh%)              ! Build an outgoing data buffer.
XXppt%=Varptr(Xbuf%(0))
XRseq%=0
XRrseq%=0
XXseq%=0
XXack%=0
XRem
XRem Since we called, we are master and send our stuff first.
XRem
XSitewrk$=Sitedir$+"\WORK"           ! Temporary work-list file.
XSitecfl$=Sitedir$+"\*.C"
XIf Exist(Sitewrk$)
X  Kill Sitewrk$                     ! Get rid of old stale one.
XEndif
XIf Exist(Sitecfl$)                  ! Do we have anything to send???
X  Dir Sitecfl$ To Sitewrk$          ! Then make a list of those files.
X  Open "I",#3,Sitewrk$
X  While Not Eof(#3)                 ! Setup to send those files, pair by pair.
X    Line Input #3,Sfil$
X    Sfil$=Sitedir$+"\"+Left$(Sfil$,Instr(Sfil$,"."))
```

```
X    Cfil$=Sfil$+"C"
X    Wfil$(1)=Sfil$+"D"
X    Wfil$(2)=Sfil$+"X"
X    If Exist(Cfil$)
X      Open "I",#4,Cfil$
X      For Wrks%=1 To 2
X        Line Input #4,Wrklin$       ! Get D. or X. worklines out of C.
X        Wrklin$=Wrklin$+Chr$(0)
X        If Exist(Wfil$(Wrks%))
X          Print "SEND:"'Wrklin$
X          Xfdone!=False
X          Xrqp%=0
X          While Not Xfdone!         ! Send the workline.
X            @Wrqtpkt
X          Wend
X          @Waitdata
X          If Peek(Rcvbf%)<>Asc("S")
X            Print "Send request failed."
X            @Hangup
X          Endif
X          If Peek(Rcvbf%+1)=Asc("Y")
X            Open "I",#1,Wfil$(Wrks%)
X            Xfl%=Lof(#1)
X            Xfp%=0
X            Xfdone!=False
X            While Not Xfdone!        ! Send D. or X. file.
X              @Wfilepkt
X            Wend
X            @Waitdata
X            If Peek(Rcvbf%)<>Asc("C") Or Peek(Rcvbf%+1)<>Asc("Y")
X              Print "Send failed."
X              @Hangup
X            Endif
X            Close #1
X          Else                       ! They won't let us do something.
X            Print "Send request refused by remote site."
X          Endif
X          Kill Wfil$(Wrks%)
X        Endif
X      Next Wrks%
X      Close #4
X      Kill Cfil$
X    Endif
X  Wend
X  Close #3
X  Kill Sitewrk$
XEndif
XRem
XRem We are done sending, now we become slave to receive.
XRem
XWrklin$="H"+Chr$(0)
XXrqp%=0
X@Wrqtpkt
```

```
X@Waitdata
XIf Peek(Rcvbf%)=Asc("H")
X  If Peek(Rcvbf%+1)=Asc("N")             ! HN means they have mail for us.
X    Do
X      Rwfr!=False
X      Wrklin$=""
X      While Not Rwfr!
X        @Waitdata                       ! Get their workline request.
X        For Wf%=0 To Rpktsiz%-1
X          Wlv%=Peek(Rcvbf%+Wf%)
X          If Wlv%=0
X            Rwfr!=True
X          Else
X            Wrklin$=Wrklin$+Chr$(Wlv%)
X          Endif
X          Exit If Rwfr!
X        Next Wf%
X      Wend
X      Rwff$=Left$(Wrklin$,1)
X      Exit If Rwff$="H"                 ! If H then they are done too.
X      Print "RCV:"'Wrklin$
X      If Rwff$="S"
X        Wlst%=Instr(5,Wrklin$," ")
X        Rwff$=Mid$(Wrklin$,Wlst%+1,1)    ! We have to shorten the file name.
X        Rfil$=Rwff$+Mid$(Wrklin$,Instr(Wlst%,Wrklin$," ")-4,4)+"."+Rwff$
X        Open "O",#1,Sitercv$+"\"+Rfil$
X        Wrklin$="SY"+Chr$(0)             ! Tell them it's okay to send.
X        Xrqp%=0
X        @Wrqtpkt
X        Rdone!=False
X        While Not Rdone!                 ! Get D. or X. file contents.
X          @Waitdata
X          If Ptyp%=&H80
X            Bput #1,Rcvbf%,Rpktsiz%
X          Else
X            Dif%=Peek(Rcvbf%)
X            Dfc%=1
X            If Dif%=>128
X              Dif%=Dif%-128+Peek(Rcvbf%+1)*128
X              Dfc%=2
X            Endif
X            Dif%=Rpktsiz%-Dif%
X            If Dif%<>0
X              Bput #1,Rcvbf%+Dfc%,Dif%
X            Else
X              Rdone!=True
X            Endif
X          Endif
X        Wend
X        Close #1
X        Wrklin$="CY"+Chr$(0)             ! Acknowledge successful copy.
X        Xrqp%=0
X        @Wrqtpkt
```

```
X      Else
X        Print "Illegal request from remote site:"
X        Print Wrklin$
X        Wrklin$=Rwff$+"N"+Chr$(0)        ! Tell them sorry, but no.
X        Xrqp%=0
X        @Wrqtpkt
X      Endif
X    Loop
X    Wrklin$="HY"+Chr$(0)
X    Xrqp%=0
X    @Wrqtpkt
X    @Waitdata
X    If Peek(Rcvbf%)<>Asc("H") Or Peek(Rcvbf%+1)<>Asc("Y")
X      Print "Failed looking to hangup."
X      @Hangup
X    Endif
X    @Fini
X  Else
X    Wrklin$="HY"+Chr$(0)
X    Xrqp%=0
X    @Wrqtpkt
X    @Fini
X  Endif
XElse
X  Print "Mode switching failed."
X  @Hangup
XEndif
XEnd
XProcedure Fini
X  Print "Done. Terminating session."
X  @Ostr("\10\09\A2\AA\08\09")
X  @Waitframe
X  If Frmcmd%<>8
X    Print "Failed waiting for CLOSE!"
X    @Hangup
X  Endif
X  @Ostr("\10OOOOOO\0A")
X  @Waitfor("OOO",5)
X  @Ostr("\10OOOOOO\0A")
X  Print "Conversation complete."
X  @Hangup
XReturn
XProcedure Ostr(O$) ! Send ASCII or HEX \XX (2 nybbles - uppercase.)
X  L%=Len(O$)
X  J%=0
X  Repeat
X    Inc J%
X    C%=Asc(Mid$(O$,J%,1))
X    If C%=&H5C
X      Inc J%
X      C%=Val("&H"+Mid$(O$,J%,2))
X      Inc J%
X    Endif
```

```
X    Out 1,C%
X  Until J%=L%
XReturn
XProcedure Waitfor(O$,Tx%) ! Look for O$ in input stream for Tx seconds.
X  Pkey!=False             ! Pressing space key aborts during dial up part.
X  Found!=False
X  Ol%=Len(O$)
X  Cp%=1
X  T%=Timer+Tx%*200
X  While Timer<T%
X    If Inp?(1)
X      C%=Inp(1) And &H7F
X      If Asc(Mid$(O$,Cp%,1))=C%
X        If Cp%=Ol%
X          Found!=True
X        Endif
X        Inc Cp%
X      Else
X        If Cp%<>1
X          Cp%=1
X          If Asc(Mid$(O$,Cp%,1))=C%
X            Inc Cp%
X          Endif
X        Endif
X      Endif
X    Endif
X    Exit If Found!
X    If Inp?(2)
X      If Inp(2)=Asc(" ")      ! Abort during login if SPACE key is pressed.
X        Pkey!=True
X      Endif
X    Endif
X    Exit If Pkey!
X  Wend
X  If ((Not Found!) And Not Retry!) Or Pkey!
X    Print "Failed waiting for: ";O$
X    @Hangup
X  Endif
XReturn
XProcedure Waitframe                 ! Wait for a six byte framing envelope.
X  Found!=False
X  Rr!=False
X  Dpkt!=False
X  Wfr%=C:Rcv%(L:Wfp%,6)
X  Repeat
X    Cntrol%=Peek(Wfp%+4)
X    Rsum1%=Peek(Wfp%+2)
X    Rsum2%=Peek(Wfp%+3)
X    Xs%=Peek(Wfp%+1) Xor Rsum1% Xor Rsum2% Xor Cntrol%
X    If Wfr%=0 And Peek(Wfp%)=&H10 And Xs%=Peek(Wfp%+5)
X      Found!=True
X      Ptyp%=Cntrol% And &HC0
X      If Ptyp%=0
```

```
X        Xack%=Cntrol% And 7
X        Frmcmd%=Cntrol% And &H38
X        If Frmcmd%=&H20
X          Rr!=True
X        Endif
X      Else
X        Dpkt!=True
X        Rrseq%=(Cntrol%/8) And 7
X        Rsum%=Rsum2%*256+Rsum1%
X      Endif
X    Else
X      If Wfr%=0
X        Bmove Wfp%+1,Wfp%,5
X        Wfr%=C:Rcv%(L:Wfp%+5,1)
X      Endif
X    Endif
X    Exit If Found!
X  Until Wfr%<>0
XReturn
XProcedure Hangup               ! Hangup.
X  Print "Hanging Up!"
X  @Ostr("\0D")
X  Pause 60
X  @Ostr("+++")
X  Pause 60
X  @Ostr("\0DATH\0D")
X  Pause 25
X  Void Xbios(30,&H10)          ! Drop DTR for 1/2 second.
X  Pause 25
X  Void Xbios(29,&HEF)          ! Raise DTR.
X  End
XReturn
XRem
XRem Writes:  One data packet, from file opened as #1, to RS232 opened as #2.
XRem Returns: Xfdone!=True after last packet sent.
XRem Initial: Xfl%=Lof(#1), Xfp%=0, Xppt%=base packet pointer,
XRem          xpktsiz%, xpktcod%, xpktsizh%.
XRem Current: Rseq%=last rcvd packet sequence number, Xseq%=last xmited.
XRem Updates: Xseq%=xmited packet sequence number, Xfp%=xmited data pointer.
XRem
XProcedure Wfilepkt
X  Xfpo%=Xfp%
X  Xfp%=Xfp%+Xpktsiz%
X  Xseq%=(Xseq%+1) And 7
X  Xph%=Xppt%+Xseq%*Xpktsizh%
X  Xpd%=Xph%+6
X  Poke Xph%,&H10
X  Poke Xph%+1,Xpktcod%
X  If Xfp%<=Xfl%
X    Bget #1,Xpd%,Xpktsiz%             ! Full Packet.
X    Xctrl%=&H80 Or Xseq%*8 Or Rseq%
X    Xsum%=C:Chk%(L:Xpd%,W:Xpktsiz%,W:Xctrl%)
X    Poke Xph%+2,Xsum%
```

```
X    Xf1%=Xsum%/256
X    Poke Xph%+3,Xf1%
X    Poke Xph%+4,Xctrl%
X    Poke Xph%+5,(Xpktcod% Xor Xsum% Xor Xf1% Xor Xctrl%)
X    @Wchkwin(0)
X  Else
X    If Xfpo%<Xfl%
X      Xbun%=Xpktsiz%-(Xfl%-Xfpo%)     ! Short Packet. (If needed.)
X      If Xbun%<128
X        Poke Xpd%,Xbun%
X        Bget #1,Xpd%+1,Xfl%-Xfpo%
X      Else
X        Poke Xpd%,Xbun% And &H7F Or &H80
X        Poke Xpd%+1,Xbun%/128
X        Bget #1,Xpd%+2,Xfl%-Xfpo%
X      Endif
X      Xctrl%=&HC0 Or Xseq%*8 Or Rseq%
X      Xsum%=C:Chk%(L:Xpd%,W:Xpktsiz%,W:Xctrl%)
X      Poke Xph%+2,Xsum%
X      Xf1%=Xsum%/256
X      Poke Xph%+3,Xf1%
X      Poke Xph%+4,Xctrl%
X      Poke Xph%+5,(Xpktcod% Xor Xsum% Xor Xf1% Xor Xctrl%)
X      @Wchkwin(0)
X    Else
X      If Xpktsiz%<128                 ! Empty Packet. (Always sent last.)
X        Poke Xpd%,Xpktsiz%
X      Else
X        Poke Xpd%,Xpktsiz% And &H7F Or &H80
X        Poke Xpd%+1,Xpktsiz%/128
X      Endif
X      Xctrl%=&HC0 Or Xseq%*8 Or Rseq%
X      Xsum%=C:Chk%(L:Xpd%,W:Xpktsiz%,W:Xctrl%)
X      Poke Xph%+2,Xsum%
X      Xf1%=Xsum%/256
X      Poke Xph%+3,Xf1%
X      Poke Xph%+4,Xctrl%
X      Poke Xph%+5,(Xpktcod% Xor Xsum% Xor Xf1% Xor Xctrl%)
X      @Wchkwin(1)
X      Xfdone!=True
X    Endif
X  Endif
XReturn
XRem
XRem This procedure sends data packets used for H,S,C type signals.
XRem It differs from the other data packets in that there are no short
XRem packets.  Instead a zero follows each command string and a zero
XRem must appear as the last byte in the packet.
XRem
XProcedure Wrqtpkt              ! Initial: Xrqp%=0, Wrklin$, Xfdone!=False.
X  Xstp%=Varptr(Wrklin$)
X  Xstl%=Len(Wrklin$)
X  Xrqpo%=Xrqp%
```

```
X  Xrqp%=Xrqp%+Xpktsiz%
X  Xseq%=(Xseq%+1) And 7
X  Xph%=Xppt%+Xseq%*Xpktsizh%
X  Xpd%=Xph%+6
X  Poke Xph%,&H10
X  Poke Xph%+1,Xpktcod%
X  Bmove Xstp%+Xrqpo%,Xpd%,Xpktsiz%
X  If Xrqp%>Xstl%
X    Poke Xpd%+Xpktsiz%-1,0
X    Xfdone!=True
X  Endif
X  Xctrl%=&H80 Or Xseq%*8 Or Rseq%
X  Xsum%=C:Chk%(L:Xpd%,W:Xpktsiz%,W:Xctrl%)
X  Poke Xph%+2,Xsum%
X  Xf1%=Xsum%/256
X  Poke Xph%+3,Xf1%
X  Poke Xph%+4,Xctrl%
X  Poke Xph%+5,(Xpktcod% Xor Xsum% Xor Xf1% Xor Xctrl%)
X  @Wchkwin(1)
XReturn
XRem
XRem This procedure sends RR's, RJ's, CLOSE, and INITA,B,C's.
XRem
XProcedure Wctlpkt(Wct%)       ! wct%=0 for RR, 1 for Rj, 2 for other.
X  If Wct%=0
X    Wcc%=&H20 Or Rseq%
X  Else
X    If Wct%=1
X      Wcc%=&H10 Or Rseq%
X    Endif
X  Endif
X  Wc1%=&HAAAA-Wcc%
X  Wc0%=Wc1% And &HFF
X  Div Wc1%,256
X  Wcx%=9 Xor Wc0% Xor Wc1% Xor Wcc%
X  Out 1,&H10
X  Out 1,&H9
X  Out 1,Wc0%
X  Out 1,Wc1%
X  Out 1,Wcc%
X  Out 1,Wcx%
XReturn
XRem
XRem This procedure waits to receive a data packet.  It RR's them as soon
XRem as it sees them.
XRem
XProcedure Waitdata
X  For Wf%=1 To 10
X    @Waitframe
X    If Dpkt!
X      Nwp%=C:Rcv%(L:Rcvbf%,Rpktsiz%)
X      Csum%=C:Chk%(L:Rcvbf%,Rpktsiz%,Cntrol%)
X      If Rsum%=Csum% And ((Rseq%+1) And 7)=Rrseq%  ! Verify data and seq.
```

```
X        Rseq%=Rrseq%
X        @Wctlpkt(0)
X      Else
X        Found!=False
X      Endif
X    Else
X      Found!=False
X    Endif
X    Exit If Found!
X    If Not Rr!                    ! Gets around a Telebit peculiarity.
X      @Wctlpkt(1)
X    Endif
X  Next Wf%
X  If Not Found!
X    Print "Failed waiting for a data frame."
X    @Hangup
X  Endif
XReturn
XRem
XRem This procedure handles the windowing of sent data packets.
XRem It sends as many packets as allowed before waiting for RR's.
XRem
XProcedure Wchkwin(Flush%)
X  If ((Xack%+Xwinsiz%+1) And 7)<>Xseq%
X    Bput #2,Xph%,Xpktsizh%
X  Else
X    For Wf%=1 To 10
X      @Waitframe
X      If Rr!
X        Bput #2,Xph%,Xpktsizh%
X      Else
X        Found!=False
X        For Jr%=1 To Xwinsiz%
X          Xjr%=(Xack%+Jr%) And 7
X          Bput #2,Xppt%+Xjr%*Xpktsizh%,Xpktsizh%
X          Exit If Xjr%=Xseq%
X        Next Jr%
X      Endif
X      Exit If Rr!
X    Next Wf%
X    If Not Found!
X      Print "Failed waiting for RR."
X      @Hangup
X    Endif
X  Endif
X  If Flush%=1
X    While Xack%<>Xseq%
X      For Wf%=1 To 10
X        @Waitframe
X        If Not Rr!
X          Found!=False
X          For Jr%=1 To Xwinsiz%
X            Xjr%=(Xack%+Jr%) And 7
```

```
X            Bput #2,Xppt%+Xjr%*Xpktsizh%,Xpktsizh%
X            Exit If Xjr%=Xseq%
X          Next Jr%
X        Endif
X        Exit If Rr!
X      Next Wf%
X      If Not Found!
X        Print "Failed waiting for flushed RR."
X        @Hangup
X      Endif
X    Wend
X  Endif
XReturn
XRem ------------------------------------------------------------------
XRem CHECKSUM ROUTINE LISTING
XRem
XRem 206F0004    start: move.l 4(sp),a0    ; data packet start address.
XRem 322F0008           move.w 8(sp),d1    ; data packet length.
XRem 322F000A           move.w 10(sp),d2   ; command byte.
XRem 203C0000FFFF       move.l #$ffff,d0
XRem 4243               clr.w d3
XRem E358        ck1:   rol.w #1,d0        ; do checksum on every
XRem 2800               move.l d0,d4       ; byte in the data packet.
XRem 4285               clr.l d5
XRem 1A18               move.b (a0)+,d5
XRem D045               add.w d5,d0
XRem 3A00               move.w d0,d5
XRem B345               eor.w d1,d5
XRem D645               add.w d5,d3
XRem B084               cmp.l d4,d0
XRem 6E02               bgt.s ovck
XRem B740               eor.w d3,d0
XRem 04410001    ovck:  sub.w #1,d1
XRem 66E4               bne.s ck1
XRem 0242FFFF           and.w #$ffff,d2
XRem B142               eor.w d0,d2       ; do checksum on command byte.
XRem 303CAAAA           move.w #$aaaa,d0
XRem 9042               sub.w d2,d0       ; subtract everything from AAAA.
XRem 02800000FFFF       and.l #$ffff,d0
XRem 4E75               rts
XRem ------------------------------------------------------------------
XRem INPUT ROUTINE LISTING
XRem
XRem 286F0004      start:  move.l 4(sp),a4     ; Buffer address.
XRem 382F0008              move.w 8(sp),d4     ; Packet size.
XRem 42A7                  clr.l -(sp)         ; Execute in superuser mode.
XRem 3F3C0020              move.w #$20,-(sp)
XRem 4E41                  trap #1
XRem 5C8F                  addq.l #6,sp
XRem 2E00                  move.l d0,d7        ; Save SUPER stack pointer.
XRem 2A39000004BA          move.l $4ba,d5      ; Get initial time.
XRem 068500000960          add.l #2400,d5      ; Twelve second timeout.
XRem 3F3C0001      next:   move.w #1,-(sp)     ; Test for character.
```

```
XRem 3F3C0001              move.w #1,-(sp)
XRem 4E4D                  trap #13
XRem 588F                  addq.l #4,sp
XRem 4A40                  tst d0
XRem 6714                  beq.s nomore        ; No character ready.
XRem 3F3C0001              move.w #1,-(sp)     ; Else, get character.
XRem 3F3C0002              move.w #2,-(sp)
XRem 4E4D                  trap #13
XRem 588F                  addq.l #4,sp
XRem 18C0                  move.b d0,(a4)+
XRem 5344                  subq.w #1,d4
XRem 670C                  beq.s out
XRem 60DC                  bra.s next
XRem 2C39000004BA  nomore: move.l $4ba,d6      ; Check timer.
XRem 9C85                  sub.l d5,d6
XRem 6DD2                  blt.s next
XRem 2F07          out:    move.l D7,-(sp)     ; Back to USER mode.
XRem 3F3C0020              move.w #$20,-(sp)
XRem 4E41                  trap #1
XRem 5C8F                  addq.l #6,sp
XRem 2004                  move.l d4,d0        ; D0=zero for success.
XRem 4E75                  rts
XRem ------------------------------------------------------------------
X
SHAR_EOF
chmod 0600 mailtruk.lst || echo "restore of mailtruk.lst fails"
sed 's/^X//' << 'SHAR_EOF' > protocol.doc &&
X                   Packet Driver Protocol
X
X                       G. L. Chesson
X                     Bell Laboratories
X                (Shortened by John Logajan)
XGeneral
X
X                            The  technique  depends  on  the
Xnotion of a transmission window to determine upper and lower
Xbounds  for  valid  sequence  numbers.   The  transmitter is
Xallowed to retransmit packets having sequence numbers within
Xthe  window  until  the receiver indicates that packets have
Xbeen correctly received.  Positive acknowledgement from  the
Xreceiver  moves  the  window; negative acknowledgement or no
Xacknowledgement causes retransmission.   The  receiver  must
Xignore  duplicate  transmission,  detect  the various errors
Xthat may occur, and inform the transmitter when packets  are
Xcorrectly or incorrectly received.
X
XPacket Formats
X
X     The protocol is defined in terms of  message  transmis-
Xsions  of  8-bit  bytes.   Each message includes one control
Xbyte plus a data segment of zero or more information  bytes.
XThe  allowed data segment sizes range between 32 and 4096 as
Xdetermined by the formula 32(2k) where k is a 3-bit  number.
```

XThe packet sequence numbers are likewise constrained to 3-
Xbits; i.e. counting proceeds modulo-8.
X
X   The control byte is partitioned into three fields as
Xdepicted below.
X
Xbit   7   6   5   4   3   2   1   0
X   t   t   x   x   x   y   y   y
X
XThe t bits indicate a packet type and determine the
Xinterpretation to be placed on the xxx and yyy fields. The
Xvarious interpretations are as follows:
X
X   tt   interpretation
X
X   00   control packet
X   10   data packet
X   11   short data packet
X
XA data segment accompanies all non-control packets. Each
Xtransmitter is constrained to observe the maximum data seg-
Xment size established during initial synchronization by the
Xreceiver that it sends to. Type 10 packets have maximal
Xsize data segments. Type 11, or short, packets have zero
Xor more data bytes but less than the maximum. The first one
Xor two bytes of the data segment of a short packet are
Xcount bytes that indicate the difference between the max-
Ximum size and the number of bytes in the short segment. If
Xthe difference is less than 127, one count byte is used. If
Xthe difference exceeds 127, then the low-order seven bits of
Xthe difference are put in the first data byte and the high-
Xorder bit is set as an indicator that the remaining bits of
Xthe difference are in the second byte.
X
X   The sequence number of a non-control packet is given by
Xthe xxx field. Control packets are not sequenced. The
Xnewest sequence number, excluding duplicate transmissions,
Xaccepted by a receiver is placed in the yyy field of non-
Xcontrol packets sent to the other receiver.
X
X   There are no data bytes associated with a control
Xpacket, the xxx field is interpreted as a control message,
Xand the yyy field is a value accompanying the control mes-
Xsage. The control messages are listed below in decreasing
Xpriority. That is, if several control messages are to be
Xsent, the lower-numbered ones are sent first.
X
X   xxx   name   yyy
X
X   1   CLOSE   n/a
X   2   RJ   last correctly received sequence number
X   3   ---   ---
X   4   RR   last correctly received sequence number

X   5   INITC   window size
X   6   INITB   data segment size
X   7   INITA   window size
X
X   The CLOSE message indicates that the communications Xchannel is to be shut down. The RJ, or reject, message Xindicates that the receiver has detected an error and the Xsender should retransmit after using the yyy field to update Xthe window. The RR, or receiver Xready, message indicates that the receiver has detected no Xerrors; the yyy field updates the sender's window. The XINITA/B/C messages are used to set window and data segment Xsizes. Segment sizes are calculated by the formula 32(2yyy) Xand window sizes may range between 1 and 7.
X
X   Measurements of the protocol running on communication Xlinks at rates up to 9600 baud showed that a window size of X2 is optimal given a packet size greater than 32 bytes. XThis means that the link bandwidth can be fully utilized by Xthe software.
X
XMessage Exchanges
X
X   Initialization
X
X   Messages are exchanged between four cooperating enti- Xties: two senders and two receivers. This means that the Xcommunication channel is thought of as two independent Xhalf-duplex data paths. For example the window and segment Xsizes need not be the same in each direction.
X
X   Initial synchronization is accomplished with two 3-way Xhandshakes: two each of INITA/INITB/INITC. Each sender Xtransmits INITA messages repeatedly. When an INITA message Xis received, INITB is sent in return. When an INITB message Xis received and an INITB message has been sent, an INITC Xmessage is sent. The INITA and INITB messages carry with Xthem the packet and window size that each receiver wants to Xuse, and the senders are supposed to comply. When a Xreceiver has seen all three INIT messages, the channel is Xconsidered to be open. INIT messages are ignored elsewhere.
X
X   Data Transport
X
X   After initial synchronization each receiver sets a Xmodulo-8 incrementing counter R to 0; each sender sets a Xsimilar counter S to 1. The value of R is always the number Xof the most recent correctly received packet. The value of XS is always the first sequence number in the output window. XLet W denote window size. Note that the value of W may be Xdifferent for each sender.
X
X   A sender may transmit packets with sequence numbers in

Xthe range S to (S+W-1) mod-8. At any particular time a Xreceiver expects arriving packets to have numbers in the Xrange (R+1) mod-8 to (R+W) mod-8. Packets must arrive in Xsequence number order are are only acknowledged in order. XThat is, the next packet a receiver will acknowledge must Xhave sequence number (R+1) mod-8.
X
X A receiver acknowledges receipt of data packets by Xarranging for the value of its R counter to be sent across Xthe channel where it will be used to update an S counter. XThis is done in two ways. If data is flowing in both direc-Xtions across a channel then each receiver's current R value Xis carried in the yyy field of non-control packets. Other-Xwise when there is no bidirectional data flow, each Xreceiver's R value is transmitted across the link as the yyy Xfield of an RR control packet.
X
X Error handling is up to the discretion of the receiver. XIt can ignore all errors in which case transmitter timeouts Xmust provide for retransmission. The receiver may also gen-Xerate RJ error control packets. The yyy field of an incom-Xing RJ message replaces the S value of the local sender and Xconstitutes a request for retransmission to start at that Xsequence number.
X
X Termination
X
X The CLOSE message is used to terminate communications. XSoftware on either or both ends of the communication channel Xmay initiate termination. In any case when one end wants to Xterminate it sends CLOSE messages until one is received from Xthe other end or until a programmable limit on the number of XCLOSE messages is reached. Receipt of a CLOSE message Xcauses a CLOSE message to be sent.
X
X Framing
X
X The term framing is used to denote the technique by Xwhich the beginning and end of a message is detected in a Xbyte stream; error control denotes the method by which Xtransmission errors are detected.
X
X A six byte framing envelope is constructed using the Xcontrol byte C of a packet and five other bytes as depicted Xbelow.
X <DLE><k><c0><c1><C><x>
XThe <DLE> symbol denotes the ASCII ctrl/P character. If the Xenvelope is to be followed by a data segment, <k> has the Xvalue log2(size)-4; i.e. 1 < k < 8. If k is 9, then the Xenvelope represents a control packet. The <c0> and <c1> Xbytes are the low-order and high-order bytes respectively of Xa 16-bit checksum. For control packets, this value is X0xAAAA minus the control byte C.

```
XFor data packets,  the  checksum  is calculated by a program.
X(John Logajan's 68000 version is below.)
XThe <x> byte is the exclusive-or of  <k><c0><c1><C>.   Error
Xcontrol  is  accomplished  by  checking  a  received framing
Xenvelope for compliance with the definition, and comparing a
Xchecksum function of the data segment with <c0><c1>.
X
X     This particular framing strategy assumes data  segments
Xare constant-sized: the  unused  bytes in a short packet are
Xactually transmitted.
X
X-------------------------------------------------------
XLogajan's 68000 version of the checksum program,
Xas called from a GFA Basic C: call.
X
Xstart: move.l 4(sp),a0      ;data packet start address
X       move.w 8(sp),d1      ;data packet length
X       move.w 10(sp),d2     ;command byte
X       move.l #$ffff,d0
X       clr.w d3
Xck1:   rol.w #1,d0
X       move.l d0,d4
X       clr.l d5
X       move.b (a0)+,d5
X       add.w d5,d0
X       move.w d0,d5
X       eor.w d1,d5
X       add.w d5,d3
X       cmp.l d4,d0
X       bgt.s ovck
X       eor.w d3,d0
Xovck:  sub.w #1,d1
X       bne.s ck1
X       and.w #$ffff,d2
X       eor.w d0,d2
X       move.w #$aaaa,d0
X       sub.w d2,d0
X       and.l #$ffff,d0      ;Checksum in D0.
X       rts
X-------------------------------------------------------
XExample session:
X
XNote below: S = we send, R = we receive.
XAfter the login: and password: stuff;
X
XR Shere[=sitename]
XS Soursitename
XR ROK
XR Pg[other protcols]
XS Ug
XS INITA (with our window size)
XR INITA (with their window size)
XS INITB (with our packet size)
```

```
XR INITB (with their packet size)
XS INITC (with our window size, again)
XR INITC
XS S workline with D.file name.   (If we have work, else H)
XR RR
XR SY
XS RR
XS data packets, short packet if need, followed by empty pkt.
XR RR's
XR CY
XS RR
XS S workline with X.file name.
XR RR
XR SY
XS data packets
XR RR's
XR CY
XS H    (If we are done)
XR RR
XR HN   (If they have work, else HY)
XS RR
XR S workline D.file
XS RR
XS SY
XR RR
XR data packets
XS RR's
XS CY
XR RR
XR S workline X.file
XS RR
XS SY
XR RR
XR data packets
XS RR's
XS CY
XR RR
XR H   (If they are done too.)
XS RR
XS HY
XR RR
XR HY
XR RR
XS CLOSE
XR CLOSE
XS OOOOOO
XR OOOOOO (Hangup)
X-----------------------------------------------------
X
SHAR_EOF
chmod 0600 protocol.doc || echo "restore of protocol.doc fails"
sed 's/^X//' << 'SHAR_EOF' > readmail.lst &&
XRem READMAIL V1.0 for use with MAILTRUK, by John Logajan, Jan '89.
```

```
XRem                        --- PUBLIC DOMAIN ---
XRem Checks mailbox and reads letters.  Saved files are always directed to the
XRem main mail directory specified below.  Please customize the next two lines.
XRem
XMdir$="c:\mail\"                ! Main upper level mail directory.
XMysite$="logajan\"              ! Name of site mailbox directory.
XRem
XMwrk$=Mdir$+Mysite$+"WORK"
XMd$=Mdir$+Mysite$+"*.*"
XIf Exist(Mwrk$)
X  Kill Mwrk$
XEndif
XIf Exist(Md$)
X  Dir Md$ To Mwrk$
X  Open "I",#3,Mwrk$
X  M=0
X  Bored!=False
X  While Not Eof(#3)
X    Line Input #3,Mfil$
X    Mfix$=Mdir$+Mysite$+Mfil$
X    If Exist(Mfix$)
X      If Right$(Mfil$,2)=".X"
X        Kill Mfix$
X      Else
X        If Mfil$<>"WORK"
X          Inc M
X          @More
X          Repeat
X            Cdn!=False
X            Line Input "Next/Command";R$
X            R$=Upper$(R$)
X            If Len(R$)=0 Or Left$(R$,1)="N"
X              Cdn!=True
X            Endif
X            If Left$(R$,1)="A" Or Left$(R$,1)="S"
X              Snam$=Mid$(R$,3)
X              If Len(Snam$)>0 And Len(Snam$)<13
X                Open "I",#4,Mfix$
X                If Left$(R$,1)="A"
X                  If Exist(Mdir$+Snam$)
X                    Open "A",#1,Mdir$+Snam$
X                  Else
X                    Print "File ";Mdir$;Snam$;" doesn't exist for appending";
X                    Print " -- Creating it."
X                    R$="S"
X                    Open "O",#1,Mdir$+Snam$
X                  Endif
X                Else
X                  Open "O",#1,Mdir$+Snam$
X                Endif
X                Lc=0
X                While Not Eof(#4)
X                  Inc Lc
```

```
X                  Line Input #4,I$
X                  Print #1,I$
X                Wend
X                Close #4
X                Close #1
X                If Left$(R$,1)="A"
X                  Print "File of ";Lc;" lines appended to ";Mdir$;Snam$
X                Else
X                  Print "File of ";Lc;" lines saved in ";Mdir$;Snam$
X                Endif
X              Else
X                Print "Filename - ";Snam$;" - too long."
X              Endif
X            Endif
X            If Left$(R$,1)="D"
X              Kill Mfix$
X              Print "Message ";M;" deleted."
X            Endif
X            If Left$(R$,1)="T"
X              If Exist(Mfix$)
X                @More
X              Else
X                Print "Sorry, you have already deleted that message."
X              Endif
X            Endif
X            If Left$(R$,1)="Q"
X              Bored!=True
X            Endif
X            Exit If Bored!
X            If Left$(R$,1)="H"
X              @Hlp
X            Endif
X          Until Cdn!
X        Endif
X      Endif
X    Endif
X    Exit If Bored!
X  Wend
X  Close #3
X  Kill Mwrk$
X  If Not Bored!
X    Print
X    Print "No other mail."
X  Endif
XElse
X  Print "No mail."
XEndif
XPause 50
XEnd
XProcedure More
X  Open "I",#4,Mfix$
X  Fll%=Lof(#4)
X  Llt%=0
```

```
X  Dne!=False
X  Print Chr$(27);"EMessage - ";M
X  Llc%=23
X  Do
X    For Lc=1 To Llc%
X      Line Input #4,C$
X      Print C$
X      Llt%=Llt%+Len(C$)
X      If Eof(#4)
X        Dne!=True
X      Endif
X      Exit If Dne!
X    Next Lc
X    Exit If Dne!
X    Pc%=Int(Llt%/Fll%*100)
X    Pmt:
X    Print Chr$(27);"p--MORE--";Chr$(27)+"q(";Pc%;"%)";
X    R%=Inp(2)
X    Print Chr$(27);"l";
X    If R%=Asc("Q") Or R%=Asc("q")
X      Dne!=True
X    Endif
X    Exit If Dne!
X    If R%=&HD
X      Llc%=1
X    Else
X      Llc%=23
X    Endif
X    If R%=Asc("T") Or R%=Asc("t")
X      Close #4
X      Open "I",#4,Mfix$
X      Fll%=Lof(#4)
X      Llt%=0
X      Print Chr$(27);"EMessage - ";M
X      Llc%=23
X    Endif
X    If R%=Asc("H") Or R%=Asc("h")
X      @Hlp
X      Goto Pmt
X    Endif
X  Loop
X  Close #4
XReturn
XProcedure Hlp
X  Print
X  Print "During message:"
X  Print "   q - skips to end."
X  Print "   t - reread from the top."
X  Print "   CR - read next line."
X  Print "   SPACE - read next page."
X  Print "   h - print this help listing."
X  Print
X  Print "At command prompt:"
```

```
X  Print "   n or CR - read next message."
X  Print "   d - delete current message."
X  Print "       (Undeleted messages remain in mailbox.)"
X  Print "   t - reread current message."
X  Print "   s filename - save message to file."
X  Print "   a filename - append message to file."
X  Print "   h - print this help listing."
X  Print "   q - quit back to desktop."
X  Print
XReturn
X
SHAR_EOF
chmod 0600 readmail.lst || echo "restore of readmail.lst fails"
sed 's/^X//' << 'SHAR_EOF' > writmail.lst &&
XRem WRITMAIL V1.0 for use with MAILTRUK  --  By John Logajan, Jan '89.
XRem                      --- PUBLIC DOMAIN ---
XRem  User configuration area.  Must be customized before using.
XRem
XUser$="john"                     ! your Mail name.
XFullname$="John Logajan"         ! your full name.
XMachine$="logajan"               ! your machine name.
XDmain$=".mn.org"                 ! your domain. (.uucp if unassigned)
XTimezone$="CST"                  ! your timezone.
XGmtfact=6                        ! your timezone to GMT factor.
XMdir$="c:\mail\"                 ! main mail directory.
XWecall$="bungia,ziltch"          ! list of machines you can call.
XSiglines=2                       ! number of lines in signature. (Below)
XDim Sig$(Siglines)
XSig$(1)="- John Logajan @ 4248 Hamline Ave; Arden Hills, MN 55112 -"
XSig$(2)="- ...rutgers!bungia!logajan!john  or john@logajan.mn.org -"
XRem
XPrint "        WRITMAIL V1.0"
XGosub Help
XRepeat
X  Line Input "Path: ",Tpath$
X  If Len(Tpath$)=0
X    End
X  Endif
X  Path$=""
X  For J=1 To Len(Tpath$)
X    If Mid$(Tpath$,J,1)<>"\" Then
X      Path$=Path$+Mid$(Tpath$,J,1)
X    Endif
X  Next J
X  P=1
X  K=Instr(Path$,"!")
X  If K=0 Then
X    Print "Path must use at least one (!) symbol."
X    P=0
X  Else
X    Neighbor$=Left$(Path$,K-1)
X    Shortpath$=Mid$(Path$,K+1)
X    If Instr(Wecall$,Neighbor$)=0 Then
```

```
X      Print "-";Neighbor$;"- is not in our list of sites that we can call."
X      P=0
X    Else
X      If Len(Shortpath$)=0 Then
X        Print "Cannot end path with an (!) symbol."
X        P=0
X      Endif
X    Endif
X  Endif
XUntil P=1
XLine Input "Subject: ",Sub$
XIf Upper$(Sub$)="~C" Then
X  End
XEndif
XDim Mes$(20000)
XM=0
XPrint
XPrint "Enter message, (.) alone sends message."
XPrint
XDo
X  If M<0 Then
X    M=0
X  Endif
X  Inc M
X  Line Input "",Mes$(M)
X  Exit If Mes$(M)="."
X  If Upper$(Mes$(M))="~C" Then
X    End
X  Endif
X  If Upper$(Mes$(M))="~H" Then
X    Dec M
X    Gosub Help
X  Endif
X  If Upper$(Mes$(M))="~D" Then
X    M=M-2
X    If M<=0 Then
X      M=0
X      Print "No message."
X    Else
X      Print
X      For P=1 To M
X        Print Mes$(P)
X      Next P
X    Endif
X  Endif
X  If Upper$(Mes$(M))="~V" Then
X    Dec M
X    If M>0 Then
X      Print
X      For P=1 To M
X        Print Mes$(P)
X      Next P
X    Else
```

```
X      Print "No message."
X    Endif
X  Endif
X  If Upper$(Left$(Mes$(M),3))="~R " Then
X    Rdfile$=Mid$(Mes$(M),4)
X    If Exist(Rdfile$) Then
X      Open "I",#1,Rdfile$
X      Lc=0
X      While Not Eof(#1)
X        Line Input #1,Mes$(M)
X        Inc M
X        Inc Lc
X      Wend
X      Close #1
X      Print "File of ";Lc;" lines read into message buffer."
X    Else
X      Dec M
X      Print "No such file!"
X    Endif
X  Endif
X  If Upper$(Mes$(M))="~S"
X    M=M-1
X    For J=1 To Siglines
X      M=M+1
X      Mes$(M)=Sig$(J)
X      Print Mes$(M)
X    Next J
X  Endif
XLoop
XIf M=1 Then
X  Print "No message to send -- Not sending."
X  Pause 50
X  End
XEndif
XSeq=10000+(Timer/100 And &H1FFF)
XSeqf$="A"+Str$(Seq And &H1FFF Or &H1000)
XDdir$=Mdir$+Neighbor$+"\"+Seqf$+".D"
XXdir$=Mdir$+Neighbor$+"\"+Seqf$+".X"
XCdir$=Mdir$+Neighbor$+"\"+Seqf$+".C"
XOpen "O",#1,Ddir$
XGosub Getdate
XYears=Year-1900
XPrint #1,"From"'Machine$;"!";User$'Dayweek$'Month$'Day'Time$'Year;Chr$(10);
XPrint #1,"Return-Path: <";Machine$;"!";User$;">";Chr$(10);
XPrint #1,"Received: by"'Machine$;Dmain$'"(MailTruk1.0)";Chr$(10);
XPrint #1,"        id AA";Seq;";"'Dayweek$;","'Day'Month$'Years'Time$'Timezone$;
XPrint #1,Chr$(10);
XPrint #1,"Date:"'Dayweek$;","'Day'Month$'Years'Time$'Timezone$;Chr$(10);
XPrint #1,"From:"'Machine$;Dmain$;"!";User$'"(";Fullname$;")";Chr$(10);
XPrint #1,"Message-Id: <";(((Years*100+Month)*100+Day)*100+Hgmt)*100+Minute;".";
XPrint #1,"AA";Seq;"@";Machine$;Dmain$;">";Chr$(10);
XPrint #1,"To:"'Path$;Chr$(10);
XIf Len(Sub$)>1 Then
```

```
X  Print #1,"Subject:"'Sub$;Chr$(10);
XEndif
XPrint #1,Chr$(10);
XFor P=1 To M-1
X  Print #1,Mes$(P);Chr$(10);
XNext P
XClose #1
XOpen "O",#2,Xdir$
XPrint #2,"U"'User$'Machine$;Chr$(10);
XPrint #2,"F"'"D.";Neighbor$;Seqf$;Chr$(10);
XPrint #2,"I"'"D.";Neighbor$;Seqf$;Chr$(10);
XPrint #2,"C rmail"'Shortpath$;Chr$(10);
XClose #2
XOpen "O",#3,Cdir$
XDest$="D."+Neighbor$+Seqf$
XBest$="D."+Machine$+Seqf$
XXest$="X."+Machine$+Seqf$
XPrint #3,"S"'Dest$'Dest$'User$'"-"'Dest$'"0666"
XPrint #3,"S"'Best$'Xest$'User$'"-"'Best$'"0666"
XClose #3
XPrint "Mail sent to outbound file area, awaiting UUCP."
XPause 50
XEnd
XProcedure Getdate            ! returns Month,Day,Year,Dayweek,Dayweek$,Month$
X  Local K,L                  ! Hour,Minute,Second,Ampm$,Hour12,Hgmt
X  Month=Val(Mid$(Date$,1,2))
X  Day=Val(Mid$(Date$,4,2))
X  Year=Val(Mid$(Date$,7,4))
X  Hour=Val(Mid$(Time$,1,2))
X  Minute=Val(Mid$(Time$,4,2))
X  Second=Val(Mid$(Time$,7,2))
X  If Hour>=12 Then
X    Ampm$="PM"
X    Hour12=Hour-12
X  Else
X    Ampm$="AM"
X    Hour12=Hour
X  Endif
X  If Hour12=0 Then
X    Hour12=12
X  Endif
X  Month$=Mid$("JanFebMarAprMayJunJulAugSepOctNovDec",Month*3-2,3)
X  Hgmt=Hour+Gmtfact
X  If Hgmt>=24 Then
X    Hgmt=Hgmt-24
X  Endif
X  K=Int(0.6+1/Month)
X  L=Year-K
X  Dayweek=Int(13*(Month+12*K+1)/5)+Int(5*L/4)-Int(L/100)+Int(L/400)+Day-1
X  Dayweek=Dayweek-(7*Int(Dayweek/7))+1
X  Dayweek$=Mid$("SunMonTueWedThuFriSat",Dayweek*3-2,3)
XReturn
XProcedure Help
```

```
X  Print
X  Print ". alone on line sends message."
X  Print "~r filename --  reads in a file."
X  Print "~d deletes previous line, and reprints message so far."
X  Print "~c cancels message."
X  Print "~h prints this help message."
X  Print "~v reprints message so far."
X  Print "~s adds predefined signature lines to message."
X  Print "Paths are of the form:  neighborsite!nextsite!...!sitex!recipient"
X  Print "Subject: is optional.  CR to skip it."
X  Print
XReturn
X
SHAR_EOF
chmod 0600 writmail.lst || echo "restore of writmail.lst fails"
exit 0


From 6369969385513490967
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,85ed7ced0f1dc817,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-11 16:47:55 PST
Path: utzoo!attcan!uunet!husc6!ogccse!blake!uw-beaver!cornell!batcomputer!
sun.soe.clarkson.edu!rpi!leah!csd4.milw.wisc.edu!bionet!agate!saturn!
ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i027: format83 -- Make Intel ASCII Hex files
Keywords: shar, C
Message-ID: <6661@saturn.ucsc.edu>
Date: 12 Mar 89 00:47:55 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 153
Approved: koreth@ssyx.ucsc.edu

Submitted-by: sun.com!sunpix!matthew (Matthew Lee Stier)
Posting-number: Volume 2, Issue 27
Archive-name: format83

This program will read a binary file and output in Intel ASCII Hex
format to standard out. This is one of the most common translation formats
for transfer to emulators and PROM programmers.

--
Matthew Lee Stier     (919) 469-8300|
Sun Microsystems ---  RTP, NC  27560|          "Wisconsin   Escapee"
uucp: {sun, rti}!sunpix!matthew     |

---- Cut Here and unpack ----
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#    Packed Mon Dec 19 15:35:52 EST 1988 by friar-taac!matthew
```

```
#    from directory /biscuits/home/matthew/IBM/Format83
#
#    Run the following text with /bin/sh to create:
#      FORMAT83.C
#      FORMAT83.DOC
#
if test -f FORMAT83.C; then echo "File FORMAT83.C exists"; else
sed 's/^X//' << 'SHAR_EOF' > FORMAT83.C &&
X/*
XAnyway, here is a simple little program which can be used on any machine
Xwith a C compiler and redirectable output (even 8 bitters!) and will work
Xwith any of the multitude of Data I/O programmers I am aware of, and most
Xlikely any programmer by any manufacturer, for that matter, since I think
Xthey all support this format (Intel Hex format).
X
XI don't deal with byte order here, but that should only be a concern if
Xyou are doing word-wide programming on a gang-banger (in which case, just
Xswap the devices in their sockets, if necessary :-) And don't forget Intel
XHex format only has a 16 bit address field, so no files > 64k are allowed.
X*/
X/*_ format83.c   Thu Sep 17 1987   Modified by: Erik Lindberg */
X/*
X      This program is released to the public domain. It can be
X      used for any purpose whatsoever. If you can get some poor
X      sucker to pay money for it, tell me your secret.
X
X      Prints a file to stdout in Intel HEX 83 format. I wrote this
X      to be easy to understand, modify, and *portable*. If you want
X      elegant, you can get there from here.
X*/
X
X#include <stdio.h>
X
X#define REC   0x10            /* Size of a record. */
X
Xchar *line, buffer[128];
XFILE *infile;
X
Xextern char hex();
X
Xmain(argc,argv)
Xint argc;
Xchar *argv[];
X{
X      int c=1, address=0;
X      int sum, i;
X      i=0;
X      if (!(infile = fopen(argv[++i],"rb"))) {
X              fprintf(stderr, "Error on open of file %s\n",argv[i]);
X              exit(1);
X      }
X
X      while (c != EOF) {
```

```
X               sum = 0;
X               line = buffer;
X               for (i=0; i<REC && (c=getc(infile)) != EOF; i++) {
X                       *line++ = hex(c>>4);
X                       *line++ = hex(c);
X                       sum += c;               /* Checksum each character. */
X               }
X               if (i) {
X                       sum += address >> 8;    /* Checksum high address byte.*/
X                       sum += address & 0xff;  /* Checksum low address byte.*/
X                       sum += i;               /* Checksum record byte count.*/
X                       line = buffer;          /* Now output the line! */
X                       putchar(':');
X                       puthex(i,2);            /* Byte count. */
X                       puthex(address,4);      /* Do address and increment */
X                       address += i;           /*    by bytes in record.   */
X                       puthex(0,2);            /* Record type.            */
X                       for(i*=2;i;i--)         /* Then the actual data.   */
X                               putchar(*line++);
X                       puthex(0-sum,2);        /* Checksum is 1byte 2's comp.*/
X                       printf("\n");
X               }
X       }
X       printf(":00000001FF\n");                /* End record. */
X}
X
X/* Return ASCII hex character for binary value. */
X
Xchar hex(c)
Xint c;
X{
X       if((c &= 0x000f)<10)
X               c += '0';
X       else
X               c += 'A'-10;
X       return((char) c);
X}
X
X/* Put specified number of digits in ASCII hex. */
X
Xputhex(val,digits)
Xint val,digits;
X{
X       if (--digits)
X               puthex(val>>4,digits);
X       putchar(hex(val & 0x0f));
X}
X
SHAR_EOF
chmod 0644 FORMAT83.C || echo "restore of FORMAT83.C fails"
fi
if test -f FORMAT83.DOC; then echo "File FORMAT83.DOC exists"; else
sed 's/^X//' << 'SHAR_EOF' > FORMAT83.DOC &&
```

```
X      This program will read a binary file and output in Intel ASCII Hex
Xformat to standard out. This is one of the most common translation formats
Xfor transfer to emulators and PROM programmers. This format can also be read
Xdirectly by the debugger supplied with both MSDOS and CPM if the extension
Xis ".HEX". I.E. if you use "format83 filename.bin> filename.hex" and then
Xload filename.hex with the debugger, you will be looking at your original
Xbinary file. If you modify the program, this is an excellent way to test
Xyour modifications.
X      This program was not written to be efficient, tricky, or elegant.
XIt was written to be easily understood, modified, and/or ported to any
Xsystem, including 8 bit systems with C compilers. Oh yeah, and it also
Xwas written because it was useful. It is free and I am releasing it to
Xthe public domain. You can do whatever you want with it. If you can get
Xsomeone to pay you for it, you have my blessing.
X      This program will not solve all your problems! One of the trickiest
Xparts about using the serial port of your garden variety PC is getting the
Xcabling and parameters right on the serial port. I have included a small
Xhex file (contents "hello world\n" :-) which I suggest you use to verify
Xyour connection with. Until you can "copy test.hex com1:" without getting
Xerrors, do not expect to be able to transfer anything meaningful to your
Xprogrammer. This program *does* work, I use it exclusively for tranferring
Xbinary files to the Data I/O programmers I use here, but it was written by
Xme personally for my own use and is not supported by Data I/O. If you do use
Xthis with a Data I/O programmer, select transfer format 83 before starting.
XIf you don't rename this program, it will impossible for you to forget :-)
SHAR_EOF
chmod 0644 FORMAT83.DOC || echo "restore of FORMAT83.DOC fails"
fi
exit 0
--
```


```
From 1111356209803460773
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,11be703faed3c14d,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-18 10:24:01 PST
Path: utzoo!attcan!uunet!wyse!vsi1!ames!pasteur!ucbvax!agate!saturn!ssyx.ucsc.edu!
koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i028: mailtruk.11 -- Version 1.1 of GFA-BASIC uucp software
Keywords: shar, GFA
Message-ID: <6716@saturn.ucsc.edu>
Date: 18 Mar 89 18:24:01 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1519
Approved: koreth@ssyx.ucsc.edu

Submitted-by: john@logajan.mn.org (John Logajan)
Posting-number: Volume 2, Issue 28
Archive-name: mailtruk.11
```

The following sources (in shar format) replace and obsolete the
previous version of the programs.

Below are five files generically named MailTruk Release V1.1.
MailTruk is a GFA Basic (2.0) implementation of uucp/uucico.
MailTruk is intended to allow non-unix/non-C Atari ST's email
access to the Usenet community. The five files included below,
in order are: MAILTRUK.DOC, MAILTRUK.LST, WRITMAIL.LST,
READMAIL.LST, and PROTOCOL.DOC.   Thanks -- John Logajan.

```
#       This is a shell archive.
#       Remove everything above and including the cut line.
#       Then run the rest of the file through sh.
#----cut here-----cut here-----cut here-----cut here----#
#!/bin/sh
# shar: Shell Archiver
#       Run the following text with /bin/sh to create:
#       mailtruk.doc
#       mailtruk.lst
#       writmail.lst
#       readmail.lst
#       protocol.doc
cat << \SHAR_EOF > mailtruk.doc
                      MailTruk Documentation
                                by
                    John Logajan, March 1989
                         4248 Hamline Ave
                       Arden Hills, MN  55112
                        john@logajan.mn.org


Note:  V1.1 seems to work on 512k machines, whereas V1.0
       apparently did not.  Both V1.1 and V1.0 work on
       1 meg machines.


INTRODUCTION TO THE NETWORK

Welcome to the world of uucp style data exchange.  Uucp stands for
Unix to Unix copy.  Unix is an operating system used by a large
number of computers.  A large network of computers running uucp
compatible data exchange has evolved over the last several years.
This network encompasses university, corporate, governmental, and
individually owned computers.  It is estimated that there are over
10,000 computers on the net, with approximately 400,000 users.
The two primary functions of the net are private electronic mail
and "broadly cast" electronic news transfer.  MailTruk is the
mechanism that transports both the news and email, however this
release of MailTruk support software does not include news reading
interfaces.

INTRODUCTION TO THE MAILTRUK, READMAIL, AND WRITMAIL PROGRAMS
```

MailTruk, ReadMail, and WritMail are uucp compatible communications programs that run on Atari ST computers. The programs are written in GFA Basic (2.0) and are freely distributed as readable listings in the public domain. Files included in the distribution are:
MAILTRUK.LST, READMAIL.LST, WRITMAIL.LST, MAILTRUK.DOC, PROTOCOL.DOC

WritMail is used to write your outgoing mail. Writmail's main task is to format your message so that other computers on the net recognize what you have sent and can figure out where it is going. Each machine, in turn, passes it along until it hopefully reaches the intended recipient.

MailTruk is the actual communications program. It first dials your neighbor site, establishes protocol agreement, and then sends the messages, if any, that you have previously prepared with WritMail. When MailTruk has finished sending, it requests to receive any mail that the remote site has pending for you. After it receives all incoming mail, MailTruk terminates the connection.

ReadMail allows you to read your mail in a convenient manner. The main objection to reading you mail files directly with GEM commands is that Unix machines use linefeed characters as end-of-line indicators rather than the carriage return character used by GEM. Thus, direct reading of mail files results in a confusing presentation.

Although MailTruk can transfer the file types used to send the Network News, additional file pre- and post-processing is required to read or write News articles. ReadMail and WritMail DO NOT handle the News format. Uncompressed News would be a fairly straight forward task to handle, but compressed and batched News would require additional sophistication. Let me know if you feel motivated to write such a program!

SETTING EVERYTHING UP

First -- You need a site to call, logon information for that site, and a modem. Since this program is being distributed over the net, if you track back from where this program came from, you're bound to run into somebody who knows your local net arrangement. (I pay $25 a year to a guy who runs a "domain park" for my access. He, in turn, calls other bigger sites -- and so on.)

Second -- All mail going to a particular site is placed in its own directory. All mail coming to your site is placed in its own directory. You should create these directories before running the programs. I am site -logajan-, I call one site -bungia-. So, I created a directory called MAIL on my C: drive where I put all my programs, and I created two sub-directories, BUNGIA and LOGAJAN, i.e. C:\MAIL\BUNGIA\

C:\MAIL\LOGAJAN\

Third -- All three programs are totally self-contained in that they do not share an external common configuration file. So you will have to insure that each of the important variables in each of the programs is correct and consistent, including the directory names mentioned above. The variables are grouped near the beginning of each program. You WILL have to customize these variables for the programs to work!

Fourth -- If you have multiple sites to call, you will have to make multiple versions of the MailTruk program (or modify it to handle multiple cases.) You will also want to write a program that recognizes messages which are just passing through, and "rewrites" the pertinent addresses, and places them back out in the outbound message area.

Fifth -- Otherwise you are a "leaf" site (you only call one machine) and any mail with your machine name in it is assumed to be to you, no matter what user name or additional path is in the To: address line.

Sixth -- The MailTruk program is designed to dail out only. If you want to re-write MailTruk to answer incoming calls -- feel free. Just remember that everthing occurs in reverse.

Seventh -- Paths; the From: and To: portion of messages; are a somewhat mysterious subject. Basically you have to know, at a minimum, the recipient's user-name and machine-name, and the path to the nearest smart mailer site. Otherwise you can specify the entire path -- but some smart sites end up sending it the route they think is best anyhow.

If I want to send to jack at atari, I enter the path as:
bungia!atari!jack Note: nearest!next!next!farthest!user
(my neighbor, bungia, knows the tortuous route to atari)

or bungia!umn-cs!rutgers!att.att.com!wally if I want to get to wally at att and know the complete path.

You will often see @'s and %'s in paths. These are Internet style paths, such as:

john@logajan.mn.org

However, note that WritMail forces you to state the path with at least one (!) symbol. Your neighbor's machine is always the first name in the path, therefore you could do:

bungia!ben@franklin.electric.com

Eighth -- Hopefully there is enough information in the listings

```
for you to set up your own site and get it operational.
                          Good luck.

SHAR_EOF
cat << \SHAR_EOF > mailtruk.lst
Rem  MAILTRUK V1.1 -- GFA Basic 2.0 / Atari ST implementation of uucp/uucico.
Rem                       --- PUBLIC DOMAIN ---
Rem  This program dials up computer sites running Unix or Unix clone uucp,
Rem      allowing the transfer of e-mail, news and other data files.
Rem
Rem       Totally original code by:  John Logajan, March 1989.
Rem                                  4248 Hamline Ave
Rem                                  Arden Hills, MN 55112
Rem      (john@logajan.mn.org  or  ...rutgers!bungia!logajan!john)
Rem
Rem    The next eight lines must be customized for each user/site.
Rem
Mysite$="logajan"              ! Your site machine name.  (You choose!)
Myuser$="logajan"              ! Your user/account name for remote site.
Mypw$="youguess"               ! Your password for remote site.
Sitedir$="\mail\bungia"        ! Directory of files to send only to that site.
Sitercv$="\mail\"+Mysite$      ! Directory for all incoming files.
Siteph$="123-4567"             ! Remote site telephone number.
Retry!=True                    ! Should we retry dialing if busy/no answer.
Void Xbios(15,7,0,&H88,1,1,-1) ! RS232 configuration/baud rate. (See below.)
Rem
Rem Configure RS232 - Xbios(15,baud,flow,ctrl,rst,xst,scr).
Rem baud = 0/19200, 1/9600, 4/2400, 7/1200, 9/300.
Rem flow = 0/None. (UUCP forbids xoff/xon! ST's RTS/CTS works goofy!)
Rem ctrl = 8bits, 1stop, noparity.
Rem rst = recv enabled.
Rem xst = xmit enabled.
Rem scr = not used.
Rem
Rwinsiz%=3           ! Number of packets in receive window. (Standard=3)
Rpktsiz%=64          ! Number of bytes in a receive packet. (Standard=64)
Rem
Rpktsizh%=Rpktsiz%+6
Rem Assembler code for calculating the checksum (speedy.)  Does the check-
Rem sum calculation on a packetsize worth of data starting at Buffer.
Rem The control byte is also included in the sum and everthing is sub-
Rem tracted from AAAA hex.
Rem            Checksum=C:Chk%(L:buffer,W:packetsize,W:controlbyte)
Rem
A$="206F0004322F0008342F000A203C0000FFFF4243E358280042851A18D0453A00B345"
A$=A$+"D645B0846E02B7400441000166E40242FFFFB142303CAAAA904202800000FFFF4E75"
Dim Chkp%(Len(A$)/4+2)
Chk%=Varptr(Chkp%(0))
For J%=0 To Len(A$)/2-1
  Poke Chk%+J%,Val("&H"+Mid$(A$,J%*2+1,2))
Next J%
Rem
Rem Assembler code for RS232 block input (speedy.)  Puts x chars into a
```

```
Rem buffer, or times out after 12 seconds.
Rem            Remainder=C:Rcv%(L:buffer,W:byteswanted)
Rem
A$="286F0004382F000842A73F3C00204E415C8F2E002A39000004BA068500000960"
A$=A$+"3F3C00013F3C00014E4D588F4A4067143F3C00013F3C00024E4D588F18C05344"
A$=A$+"670C60DC2C39000004BA9C856DD22F073F3C00204E415C8F20044E75"
Dim Rcvp%(Len(A$)/4+2)
Rcv%=Varptr(Rcvp%(0))
For J%=0 To Len(A$)/2-1
  Poke Rcv%+J%,Val("&H"+Mid$(A$,J%*2+1,2))
Next J%
Dim Rcxx%(Rpktsiz%/4+2)             ! Build an internal input buffer.
Rcvbf%=Varptr(Rcxx%(0))
Dim Wfil$(2)
Wfm=0
Wfp%=Varptr(Wfm)                    ! Build a frame buffer.
Rs232p%=Xbios(14,0)                 ! Get RS232 TOS buffer info pointer.
Rinsiz%=((Rpktsiz%+6)*Rwinsiz%+128)
Dim Rin%(Rinsiz%/4+1)               ! Make a large RS232 TOS input buffer.
Lpoke Rs232p%,Varptr(Rin%(0))
Dpoke Rs232p%+4,Rinsiz%
Dpoke Rs232p%+6,0                   ! Flag it empty.
Dpoke Rs232p%+8,0
Open "",#2,"AUX:"                   ! Open RS232 port as file #2.
Rem
Rem   You will probably need to customize the dialing part of the program,
Rem   depending upon the quirks of the site you are trying to call, and
Rem   your modem response codes.
Rem
Print "Dialing.       Push SPACE key to abort connection attempt."
Void Xbios(15,-1,-1,-1,-1,9,-1)  !  Send a break for Telebit autobaud reset.
Pause 20
Void Xbios(15,-1,-1,-1,-1,1,-1)
Pause 20
Repeat
  @Ostr("ATDT"+Siteph$+"\0D")                  !  Dial remote site.
  @Waitfor("CONNECT",60)
  If Not Found!
    Print "Busy/no answer. Dialing again."
  Endif
Until Found!
Print "Connected."
Retry!=False
Pause 200
Out 1,&HD                            ! Send a couple of CR's to prime remote.
Pause 50
Out 1,&HD
Pause 50
Out 1,&HD
Pause 50
Out 1,&HD
@Waitfor("ogin:",30)
Pause 50
```

```
@Ostr(Myuser$+"\0D")
@Waitfor("sword:",30)
@Ostr(Mypw$+"\0D")
@Waitfor("Shere",30)
Print "Logged in."
Rem
Rem Initialization sequence.
Rem
@Ostr("\10S"+Mysite$+"\0A")
@Waitfor("ROK",30)
@Waitfor("P",30)
@Waitfor("g",5)
@Ostr("\10Ug\0A")
Inita!=False
Initb!=False
Initc!=False
For Wf%=1 To 10                 ! INIT with window size / data segement size.
  If Inita!=False Or Initb!=False
    Wcc%=&H38 Or Rwinsiz%
    @Wctlpkt(2)                       ! inita window size.
  Endif
  @Waitframe
  If Found! And (Cntrol% And &HF8)=&H38
    Inita!=True
    Xwinsiz%=Cntrol% And 7
    Rpktcod%=Int(Log(Rpktsiz%)/Log(2)-4.9)
    Wcc%=&H30 Or Rpktcod%
    @Wctlpkt(2)                       ! initb packet size.
    @Waitframe
  Endif
  If Found! And (Cntrol% And &HF8)=&H30
    Initb!=True
    Xpktcod%=(Cntrol% And 7)+1
    Xpktsiz%=32*2^(Xpktcod%-1)
    Xpktsizh%=Xpktsiz%+6
  Endif
  Exit If Inita! And Initb!
Next Wf%
If Initb!
  Wcc%=&H28 Or Rwinsiz%
  @Wctlpkt(2)                       ! initc window size.
  For Wf%=1 To 3
    @Waitframe
    If Found! And (Cntrol% And &HF8)=&H28
      Initc!=True
    Endif
    Exit If Initc!
  Next Wf%
Endif
If Inita!=False Or Initb!=False Or Initc!=False
  Print "Failed initilization."
  @Hangup
Endif
```

```
Print "They want a sending window size = ";Xwinsiz%
Print "They want a sending data packet size = ";Xpktsiz%
Dim Xbuf%(2*Xpktsizh%)              ! Build an outgoing data buffer.
Xppt%=Varptr(Xbuf%(0))
Rseq%=0
Rrseq%=0
Xseq%=0
Xack%=0
Rem
Rem Since we called, we are master and send our stuff first.
Rem
Sitewrk$=Sitedir$+"\WORK"           ! Temporary work-list file.
Sitecfl$=Sitedir$+"\*.C"
If Exist(Sitewrk$)
  Kill Sitewrk$                     ! Get rid of old stale one.
Endif
If Exist(Sitecfl$)                  ! Do we have anything to send???
  Dir Sitecfl$ To Sitewrk$          ! Then make a list of those files.
  Open "I",#3,Sitewrk$
  While Not Eof(#3)                 ! Setup to send those files, pair by pair.
    Line Input #3,Sfil$
    Sfil$=Sitedir$+"\"+Left$(Sfil$,Instr(Sfil$,"."))
    Cfil$=Sfil$+"C"
    Wfil$(1)=Sfil$+"D"
    Wfil$(2)=Sfil$+"X"
    If Exist(Cfil$)
      Open "I",#4,Cfil$
      For Wrks%=1 To 2
        Line Input #4,Wrklin$       ! Get D. or X. worklines out of C.
        Wrklin$=Wrklin$+Chr$(0)
        If Exist(Wfil$(Wrks%))
          Print "SEND:"'Wrklin$
          Xfdone!=False
          Xrqp%=0
          While Not Xfdone!         ! Send the workline.
            @Wrqtpkt
          Wend
          @Waitdata
          If Peek(Rcvbf%)<>Asc("S")
            Print "Send request failed."
            @Hangup
          Endif
          If Peek(Rcvbf%+1)=Asc("Y")
            Open "I",#1,Wfil$(Wrks%)
            Xfl%=Lof(#1)
            Xfp%=0
            Xfdone!=False
            While Not Xfdone!        ! Send D. or X. file.
              @Wfilepkt
            Wend
            @Waitdata
            If Peek(Rcvbf%)<>Asc("C") Or Peek(Rcvbf%+1)<>Asc("Y")
              Print "Send failed."
```

```
            @Hangup
          Endif
          Close #1
        Else                        ! They won't let us do something.
          Print "Send request refused by remote site."
        Endif
        Kill Wfil$(Wrks%)
      Endif
    Next Wrks%
    Close #4
    Kill Cfil$
  Endif
 Wend
 Close #3
 Kill Sitewrk$
Endif
Rem
Rem We are done sending, now we become slave to receive.
Rem
Wrklin$="H"+Chr$(0)
Xrqp%=0
@Wrqtpkt
@Waitdata
If Peek(Rcvbf%)=Asc("H")
 If Peek(Rcvbf%+1)=Asc("N")             ! HN means they have mail for us.
  Do
    Wrklin$=String$(128," ")
    Wfz%=Varptr(Wrklin$)
    Wlv%=0
    Repeat
      @Waitdata                         ! Get their workline request.
      Bmove Rcvbf%,Wfz%+Wlv%,Rpktsiz%
      Wlv%=Wlv%+Rpktsiz%
      Wllv%=Instr(Wrklin$,Chr$(0))
    Until Wllv%<>0
    Wrklin$=Left$(Wrklin$,Wllv%)
    Rwff$=Left$(Wrklin$,1)
    Exit If Rwff$="H"                   ! If H then they are done too.
    Print "RCV:"'Wrklin$
    If Rwff$="S"
      Wlst%=Instr(5,Wrklin$," ")
      Rwff$=Mid$(Wrklin$,Wlst%+1,1)     ! We have to shorten the file name.
      Rfil$=Rwff$+Mid$(Wrklin$,Instr(Wlst%,Wrklin$," ")-4,4)+"."+Rwff$
      Open "O",#1,Sitercv$+"\"+Rfil$
      Wrklin$="SY"+Chr$(0)              ! Tell them it's okay to send.
      Xrqp%=0
      @Wrqtpkt
      Rdone!=False
      While Not Rdone!                  ! Get D. or X. file contents.
        @Waitdata
        If Ptyp%=&H80
          Bput #1,Rcvbf%,Rpktsiz%
        Else
```

```
            Dif%=Peek(Rcvbf%)
            Dfc%=1
            If Dif%=>128
              Dif%=Dif%-128+Peek(Rcvbf%+1)*128
              Dfc%=2
            Endif
            Dif%=Rpktsiz%-Dif%
            If Dif%<>0
              Bput #1,Rcvbf%+Dfc%,Dif%
            Else
              Rdone!=True
            Endif
          Endif
        Wend
        Close #1
        Wrklin$="CY"+Chr$(0)             ! Acknowledge successful copy.
        Xrqp%=0
        @Wrqtpkt
      Else
        Print "Illegal request from remote site:"
        Print Wrklin$
        Wrklin$=Rwff$+"N"+Chr$(0)        ! Tell them sorry, but no.
        Xrqp%=0
        @Wrqtpkt
      Endif
    Loop
    Wrklin$="HY"+Chr$(0)
    Xrqp%=0
    @Wrqtpkt
    @Waitdata
    If Peek(Rcvbf%)<>Asc("H") Or Peek(Rcvbf%+1)<>Asc("Y")
      Print "Failed looking to hangup."
      @Hangup
    Endif
    @Fini
  Else
    Wrklin$="HY"+Chr$(0)
    Xrqp%=0
    @Wrqtpkt
    @Fini
  Endif
Else
  Print "Mode switching failed."
  @Hangup
Endif
End
Procedure Fini
  Print "Done. Terminating session."
  @Ostr("\10\09\A2\AA\08\09")
  @Waitframe
  If Frmcmd%=32
    @Waitframe
  Endif
```

```
  If Frmcmd%<>8
    Print "Failed waiting for CLOSE! "
    @Hangup
  Endif
  @Ostr("\10OOOOOO\0A")
  @Waitfor("OOO",5)
  @Ostr("\10OOOOOO\0A")
  Print "Conversation complete."
  @Hangup
Return
Procedure Ostr(O$) ! Send ASCII or HEX \XX (2 nybbles - uppercase.)
  L%=Len(O$)
  J%=0
  Repeat
    Inc J%
    C%=Asc(Mid$(O$,J%,1))
    If C%=&H5C
      Inc J%
      C%=Val("&H"+Mid$(O$,J%,2))
      Inc J%
    Endif
    Out 1,C%
  Until J%=L%
Return
Procedure Waitfor(O$,Tx%) ! Look for O$ in input stream for Tx seconds.
  Pkey!=False             ! Pressing space key aborts during dial up part.
  Found!=False
  Ol%=Len(O$)
  Cp%=1
  T%=Timer+Tx%*200
  While Timer<T%
    If Inp?(1)
      C%=Inp(1) And &H7F
      If Asc(Mid$(O$,Cp%,1))=C%
        If Cp%=Ol%
          Found!=True
        Endif
        Inc Cp%
      Else
        If Cp%<>1
          Cp%=1
          If Asc(Mid$(O$,Cp%,1))=C%
            Inc Cp%
          Endif
        Endif
      Endif
    Endif
    Exit If Found!
    If Inp?(2)
      If Inp(2)=Asc(" ")      ! Abort during login if SPACE key is pressed.
        Pkey!=True
      Endif
    Endif
```

```
    Exit If Pkey!
  Wend
  If ((Not Found!) And (Not Retry!)) Or Pkey!
    Print "Failed waiting for: ";O$
    @Hangup
  Endif
Return
Procedure Waitframe                ! Wait for a six byte framing envelope.
  Found!=False
  Rr!=False
  Dpkt!=False
  Wfr%=C:Rcv%(L:Wfp%,6)
  Repeat
    Cntrol%=Peek(Wfp%+4)
    Rsum1%=Peek(Wfp%+2)
    Rsum2%=Peek(Wfp%+3)
    Xs%=Peek(Wfp%+1) Xor Rsum1% Xor Rsum2% Xor Cntrol%
    If Wfr%=0 And Peek(Wfp%)=&H10 And Xs%=Peek(Wfp%+5)
      Found!=True
      Ptyp%=Cntrol% And &HC0
      If Ptyp%=0
        Xack%=Cntrol% And 7
        Frmcmd%=Cntrol% And &H38
        If Frmcmd%=&H20
          Rr!=True
        Endif
      Else
        Dpkt!=True
        Rrseq%=(Cntrol%/8) And 7
        Rsum%=Rsum2%*256+Rsum1%
      Endif
    Else
      If Wfr%=0
        Bmove Wfp%+1,Wfp%,5
        Wfr%=C:Rcv%(L:Wfp%+5,1)
      Endif
    Endif
    Exit If Found!
  Until Wfr%<>0
Return
Procedure Hangup               ! Hangup.
  Print "Hanging Up!"
  @Ostr("\0D")
  Pause 60
  @Ostr("+++")
  Pause 60
  @Ostr("\0DATH\0D")
  Pause 25
  Void Xbios(30,&H10)          ! Drop DTR for 1/2 second.
  Pause 25
  Void Xbios(29,&HEF)          ! Raise DTR.
  End
Return
```

```
Rem
Rem Writes:  One data packet, from file opened as #1, to RS232 opened as #2.
Rem Returns: Xfdone!=True after last packet sent.
Rem Initial: Xfl%=Lof(#1), Xfp%=0, Xppt%=base packet pointer,
Rem          xpktsiz%, xpktcod%, xpktsizh%.
Rem Current: Rseq%=last rcvd packet sequence number, Xseq%=last xmited.
Rem Updates: Xseq%=xmited packet sequence number, Xfp%=xmited data pointer.
Rem
Procedure Wfilepkt
  Xfpo%=Xfp%
  Xfp%=Xfp%+Xpktsiz%
  Xseq%=(Xseq%+1) And 7
  Xph%=Xppt%+Xseq%*Xpktsizh%
  Xpd%=Xph%+6
  Poke Xph%,&H10
  Poke Xph%+1,Xpktcod%
  If Xfp%<=Xfl%
    Bget #1,Xpd%,Xpktsiz%              ! Full Packet.
    Xctrl%=&H80 Or Xseq%*8 Or Rseq%
    Xsum%=C:Chk%(L:Xpd%,W:Xpktsiz%,W:Xctrl%)
    Poke Xph%+2,Xsum%
    Xf1%=Xsum%/256
    Poke Xph%+3,Xf1%
    Poke Xph%+4,Xctrl%
    Poke Xph%+5,(Xpktcod% Xor Xsum% Xor Xf1% Xor Xctrl%)
    @Wchkwin(0)
  Else
    If Xfpo%<Xfl%
      Xbun%=Xpktsiz%-(Xfl%-Xfpo%)      ! Short Packet. (If needed.)
      If Xbun%<128
        Poke Xpd%,Xbun%
        Bget #1,Xpd%+1,Xfl%-Xfpo%
      Else
        Poke Xpd%,Xbun% And &H7F Or &H80
        Poke Xpd%+1,Xbun%/128
        Bget #1,Xpd%+2,Xfl%-Xfpo%
      Endif
      Xctrl%=&HC0 Or Xseq%*8 Or Rseq%
      Xsum%=C:Chk%(L:Xpd%,W:Xpktsiz%,W:Xctrl%)
      Poke Xph%+2,Xsum%
      Xf1%=Xsum%/256
      Poke Xph%+3,Xf1%
      Poke Xph%+4,Xctrl%
      Poke Xph%+5,(Xpktcod% Xor Xsum% Xor Xf1% Xor Xctrl%)
      @Wchkwin(0)
    Else
      If Xpktsiz%<128                  ! Empty Packet. (Always sent last.)
        Poke Xpd%,Xpktsiz%
      Else
        Poke Xpd%,Xpktsiz% And &H7F Or &H80
        Poke Xpd%+1,Xpktsiz%/128
      Endif
      Xctrl%=&HC0 Or Xseq%*8 Or Rseq%
```

```
      Xsum%=C:Chk%(L:Xpd%,W:Xpktsiz%,W:Xctrl%)
      Poke Xph%+2,Xsum%
      Xf1%=Xsum%/256
      Poke Xph%+3,Xf1%
      Poke Xph%+4,Xctrl%
      Poke Xph%+5,(Xpktcod% Xor Xsum% Xor Xf1% Xor Xctrl%)
      @Wchkwin(1)
      Xfdone!=True
    Endif
  Endif
Return
Rem
Rem This procedure sends data packets used for H,S,C type signals.
Rem It differs from the other data packets in that there are no short
Rem packets.  Instead a zero follows each command string and a zero
Rem must appear as the last byte in the packet.
Rem
Procedure Wrqtpkt             ! Initial: Xrqp%=0, Wrklin$, Xfdone!=False.
  Xstp%=Varptr(Wrklin$)
  Xstl%=Len(Wrklin$)
  Xrqpo%=Xrqp%
  Xrqp%=Xrqp%+Xpktsiz%
  Xseq%=(Xseq%+1) And 7
  Xph%=Xppt%+Xseq%*Xpktsizh%
  Xpd%=Xph%+6
  Poke Xph%,&H10
  Poke Xph%+1,Xpktcod%
  Bmove Xstp%+Xrqpo%,Xpd%,Xpktsiz%
  If Xrqp%>Xstl%
    Poke Xpd%+Xpktsiz%-1,0
    Xfdone!=True
  Endif
  Xctrl%=&H80 Or Xseq%*8 Or Rseq%
  Xsum%=C:Chk%(L:Xpd%,W:Xpktsiz%,W:Xctrl%)
  Poke Xph%+2,Xsum%
  Xf1%=Xsum%/256
  Poke Xph%+3,Xf1%
  Poke Xph%+4,Xctrl%
  Poke Xph%+5,(Xpktcod% Xor Xsum% Xor Xf1% Xor Xctrl%)
  @Wchkwin(1)
Return
Rem
Rem This procedure sends RR's, RJ's, CLOSE, and INITA,B,C's.
Rem
Procedure Wctlpkt(Wct%)       ! wct%=0 for RR, 1 for Rj, 2 for other.
  If Wct%=0
    Wcc%=&H20 Or Rseq%
  Else
    If Wct%=1
      Wcc%=&H10 Or Rseq%
    Endif
  Endif
  Wc1%=&HAAAA-Wcc%
```

```
  Wc0%=Wc1% And &HFF
  Div Wc1%,256
  Wcx%=9 Xor Wc0% Xor Wc1% Xor Wcc%
  Out 1,&H10
  Out 1,&H9
  Out 1,Wc0%
  Out 1,Wc1%
  Out 1,Wcc%
  Out 1,Wcx%
Return
Rem
Rem This procedure waits to receive a data packet.  It RR's them as soon
Rem as it sees them.
Rem
Procedure Waitdata
  For Wf%=1 To 10
    @Waitframe
    If Dpkt!
      Nwp%=C:Rcv%(L:Rcvbf%,Rpktsiz%)
      Csum%=C:Chk%(L:Rcvbf%,Rpktsiz%,Cntrol%)
      If Rsum%=Csum% And ((Rseq%+1) And 7)=Rrseq%  ! Verify data and seq.
        Rseq%=Rrseq%
        @Wctlpkt(0)
      Else
        Found!=False
      Endif
    Else
      Found!=False
    Endif
    Exit If Found!
    If Not Rr!                     ! Gets around a Telebit peculiarity.
      @Wctlpkt(1)
    Endif
  Next Wf%
  If Not Found!
    Print "Failed waiting for a data frame."
    @Hangup
  Endif
Return
Rem
Rem This procedure handles the windowing of sent data packets.
Rem It sends as many packets as allowed before waiting for RR's.
Rem
Procedure Wchkwin(Flush%)
  If ((Xack%+Xwinsiz%+1) And 7)<>Xseq%
    Bput #2,Xph%,Xpktsizh%
  Else
    For Wf%=1 To 10
      @Waitframe
      If Rr!
        Bput #2,Xph%,Xpktsizh%
      Else
        Found!=False
```

```
        For Jr%=1 To Xwinsiz%
          Xjr%=(Xack%+Jr%) And 7
          Bput #2,Xppt%+Xjr%*Xpktsizh%,Xpktsizh%
          Exit If Xjr%=Xseq%
        Next Jr%
      Endif
      Exit If Rr!
    Next Wf%
    If Not Found!
      Print "Failed waiting for RR."
      @Hangup
    Endif
  Endif
  If Flush%=1
    While Xack%<>Xseq%
      For Wf%=1 To 10
        @Waitframe
        If Not Rr!
          Found!=False
          For Jr%=1 To Xwinsiz%
            Xjr%=(Xack%+Jr%) And 7
            Bput #2,Xppt%+Xjr%*Xpktsizh%,Xpktsizh%
            Exit If Xjr%=Xseq%
          Next Jr%
        Endif
        Exit If Rr!
      Next Wf%
      If Not Found!
        Print "Failed waiting for flushed RR."
        @Hangup
      Endif
    Wend
  Endif
Return
Rem ------------------------------------------------------------------
Rem CHECKSUM ROUTINE LISTING
Rem
Rem 206F0004    start: move.l 4(sp),a0    ; data packet start address.
Rem 322F0008           move.w 8(sp),d1    ; data packet length.
Rem 342F000A           move.w 10(sp),d2   ; command byte.
Rem 203C0000FFFF       move.l #$ffff,d0
Rem 4243               clr.w d3
Rem E358        ck1:   rol.w #1,d0        ; do checksum on every
Rem 2800               move.l d0,d4       ; byte in the data packet.
Rem 4285               clr.l d5
Rem 1A18               move.b (a0)+,d5
Rem D045               add.w d5,d0
Rem 3A00               move.w d0,d5
Rem B345               eor.w d1,d5
Rem D645               add.w d5,d3
Rem B084               cmp.l d4,d0
Rem 6E02               bgt.s ovck
Rem B740               eor.w d3,d0
```

```
Rem 04410001    ovck:  sub.w #1,d1
Rem 66E4               bne.s ck1
Rem 0242FFFF           and.w #$ffff,d2
Rem B142               eor.w d0,d2       ; do checksum on command byte.
Rem 303CAAAA           move.w #$aaaa,d0
Rem 9042               sub.w d2,d0       ; subtract everything from AAAA.
Rem 02800000FFFF       and.l #$ffff,d0
Rem 4E75               rts
Rem ------------------------------------------------------------------
Rem INPUT ROUTINE LISTING
Rem
Rem 286F0004      start:  move.l 4(sp),a4     ; Buffer address.
Rem 382F0008              move.w 8(sp),d4     ; Packet size.
Rem 42A7                  clr.l -(sp)         ; Execute in superuser mode.
Rem 3F3C0020              move.w #$20,-(sp)
Rem 4E41                  trap #1
Rem 5C8F                  addq.l #6,sp
Rem 2E00                  move.l d0,d7        ; Save SUPER stack pointer.
Rem 2A39000004BA          move.l $4ba,d5      ; Get initial time.
Rem 068500000960          add.l #2400,d5      ; Twelve second timeout.
Rem 3F3C0001      next:   move.w #1,-(sp)     ; Test for character.
Rem 3F3C0001              move.w #1,-(sp)
Rem 4E4D                  trap #13
Rem 588F                  addq.l #4,sp
Rem 4A40                  tst d0
Rem 6714                  beq.s nomore        ; No character ready.
Rem 3F3C0001              move.w #1,-(sp)     ; Else, get character.
Rem 3F3C0002              move.w #2,-(sp)
Rem 4E4D                  trap #13
Rem 588F                  addq.l #4,sp
Rem 18C0                  move.b d0,(a4)+
Rem 5344                  subq.w #1,d4
Rem 670C                  beq.s out
Rem 60DC                  bra.s next
Rem 2C39000004BA  nomore: move.l $4ba,d6      ; Check timer.
Rem 9C85                  sub.l d5,d6
Rem 6DD2                  blt.s next
Rem 2F07          out:    move.l D7,-(sp)     ; Back to USER mode.
Rem 3F3C0020              move.w #$20,-(sp)
Rem 4E41                  trap #1
Rem 5C8F                  addq.l #6,sp
Rem 2004                  move.l d4,d0        ; D0=zero for success.
Rem 4E75                  rts
Rem ------------------------------------------------------------------
SHAR_EOF
cat << \SHAR_EOF > writmail.lst
Rem WRITMAIL V1.1 for use with MAILTRUK  --  By John Logajan, Jan '89.
Rem                      --- PUBLIC DOMAIN ---
Rem  User configuration area.  Must be customized before using.
Rem
User$="john"                    ! your Mail name.
Fullname$="John Logajan"        ! your full name.
Machine$="logajan"              ! your machine name.
```

```
Dmain$=".mn.org"                ! your domain. (.uucp if unassigned)
Timezone$="CST"                 ! your timezone.
Gmtfact=6                       ! your timezone to GMT factor.
Mdir$="c:\mail\"                ! main mail directory.
Wecall$="bungia,ziltch"         ! list of machines you can call.
Siglines=2                      ! number of lines in signature. (Below)
Dim Sig$(Siglines)
Sig$(1)="- John Logajan @ 4248 Hamline Ave; Arden Hills, MN 55112 -"
Sig$(2)="- ...rutgers!bungia!logajan!john  or john@logajan.mn.org -"
Rem
Print "        WRITMAIL V1.1"
Gosub Help
Repeat
  Line Input "Path: ",Tpath$
  If Len(Tpath$)=0
    End
  Endif
  Path$=""
  For J=1 To Len(Tpath$)
    If Mid$(Tpath$,J,1)<>"\" Then
      Path$=Path$+Mid$(Tpath$,J,1)
    Endif
  Next J
  P=1
  K=Instr(Path$,"!")
  If K=0 Then
    Print "Path must use at least one (!) symbol."
    P=0
  Else
    Neighbor$=Left$(Path$,K-1)
    Shortpath$=Mid$(Path$,K+1)
    If Instr(Wecall$,Neighbor$)=0 Then
      Print "-";Neighbor$;"- is not in our list of sites that we can call."
      P=0
    Else
      If Len(Shortpath$)=0 Then
        Print "Cannot end path with an (!) symbol."
        P=0
      Endif
    Endif
  Endif
Until P=1
Line Input "Subject: ",Sub$
If Upper$(Sub$)="~C" Then
  End
Endif
Dim Mes$(20000)
M=0
Print
Print "Enter message, (.) alone sends message."
Print
Do
  If M<0 Then
```

```
  M=0
Endif
Inc M
Line Input "",Mes$(M)
Exit If Mes$(M)="."
If Upper$(Mes$(M))="~C" Then
  End
Endif
If Upper$(Mes$(M))="~H" Then
  Dec M
  Gosub Help
Endif
If Upper$(Mes$(M))="~D" Then
  M=M-2
  If M<=0 Then
    M=0
    Print "No message."
  Else
    Print
    For P=1 To M
      Print Mes$(P)
    Next P
  Endif
Endif
If Upper$(Mes$(M))="~V" Then
  Dec M
  If M>0 Then
    Print
    For P=1 To M
      Print Mes$(P)
    Next P
  Else
    Print "No message."
  Endif
Endif
If Upper$(Left$(Mes$(M),3))="~R " Then
  Rdfile$=Mid$(Mes$(M),4)
  If Exist(Rdfile$) Then
    Open "I",#1,Rdfile$
    Lc=0
    While Not Eof(#1)
      Line Input #1,Mes$(M)
      Inc M
      Inc Lc
    Wend
    Close #1
    Print "File of ";Lc;" lines read into message buffer."
  Else
    Dec M
    Print "No such file!"
  Endif
Endif
If Upper$(Mes$(M))="~S"
```

```
    M=M-1
    For J=1 To Siglines
      M=M+1
      Mes$(M)=Sig$(J)
      Print Mes$(M)
    Next J
  Endif
Loop
If M=1 Then
  Print "No message to send -- Not sending."
  Pause 50
  End
Endif
Seq=10000+(Timer/100 And &H1FFF)
Seqf$="A"+Str$(Seq And &H1FFF Or &H1000)
Ddir$=Mdir$+Neighbor$+"\"+Seqf$+".D"
Xdir$=Mdir$+Neighbor$+"\"+Seqf$+".X"
Cdir$=Mdir$+Neighbor$+"\"+Seqf$+".C"
Open "O",#1,Ddir$
Gosub Getdate
Years=Year-1900
Print #1,"From"'Machine$;"!";User$'Dayweek$'Month$'Day'Time$'Year;Chr$(10);
Print #1,"Return-Path: <";Machine$;"!";User$;">";Chr$(10);
Print #1,"Received: by"'Machine$;Dmain$'"(MailTruk1.1)";Chr$(10);
Print #1,"        id AA";Seq;";"'Dayweek$;","'Day'Month$'Years'Time$'Timezone$;
Print #1,Chr$(10);
Print #1,"Date:"'Dayweek$;","'Day'Month$'Years'Time$'Timezone$;Chr$(10);
Print #1,"From:"'Machine$;Dmain$;"!";User$'"(";Fullname$;")";Chr$(10);
Print #1,"Message-Id: <";(((Years*100+Month)*100+Day)*100+Hgmt)*100+Minute;".";
Print #1,"AA";Seq;"@";Machine$;Dmain$;">";Chr$(10);
Print #1,"To:"'Path$;Chr$(10);
If Len(Sub$)>1 Then
  Print #1,"Subject:"'Sub$;Chr$(10);
Endif
Print #1,Chr$(10);
For P=1 To M-1
  Print #1,Mes$(P);Chr$(10);
Next P
Close #1
Open "O",#2,Xdir$
Print #2,"U"'User$'Machine$;Chr$(10);
Print #2,"F"'"D.";Neighbor$;Seqf$;Chr$(10);
Print #2,"I"'"D.";Neighbor$;Seqf$;Chr$(10);
Print #2,"C rmail"'Shortpath$;Chr$(10);
Close #2
Open "O",#3,Cdir$
Dest$="D."+Neighbor$+Seqf$
Best$="D."+Machine$+Seqf$
Xest$="X."+Machine$+Seqf$
Print #3,"S"'Dest$'Dest$'User$'"-"'Dest$'"0666"
Print #3,"S"'Best$'Xest$'User$'"-"'Best$'"0666"
Close #3
Print "Mail sent to outbound file area, waiting for the MailTruk."
```

```
Pause 50
End
Procedure Getdate            ! returns Month,Day,Year,Dayweek,Dayweek$,Month$
  Local K,L                  ! Hour,Minute,Second,Ampm$,Hour12,Hgmt
  Month=Val(Mid$(Date$,1,2))
  Day=Val(Mid$(Date$,4,2))
  Year=Val(Mid$(Date$,7,4))
  Hour=Val(Mid$(Time$,1,2))
  Minute=Val(Mid$(Time$,4,2))
  Second=Val(Mid$(Time$,7,2))
  If Hour>=12 Then
    Ampm$="PM"
    Hour12=Hour-12
  Else
    Ampm$="AM"
    Hour12=Hour
  Endif
  If Hour12=0 Then
    Hour12=12
  Endif
  Month$=Mid$("JanFebMarAprMayJunJulAugSepOctNovDec",Month*3-2,3)
  Hgmt=Hour+Gmtfact
  If Hgmt>=24 Then
    Hgmt=Hgmt-24
  Endif
  K=Int(0.6+1/Month)
  L=Year-K
  Dayweek=Int(13*(Month+12*K+1)/5)+Int(5*L/4)-Int(L/100)+Int(L/400)+Day-1
  Dayweek=Dayweek-(7*Int(Dayweek/7))+1
  Dayweek$=Mid$("SunMonTueWedThuFriSat",Dayweek*3-2,3)
Return
Procedure Help
  Print
  Print ". alone on line sends message."
  Print "~r filename --  reads in a file."
  Print "~d deletes previous line, and reprints message so far."
  Print "~c cancels message."
  Print "~h prints this help message."
  Print "~v reprints message so far."
  Print "~s adds predefined signature lines to message."
  Print "Paths are of the form:  neighborsite!nextsite!...!sitex!recipient"
  Print "Subject: is optional.  CR to skip it."
  Print
Return
SHAR_EOF
cat << \SHAR_EOF > readmail.lst
Rem READMAIL V1.1 for use with MAILTRUK, by John Logajan, Jan '89.
Rem                        --- PUBLIC DOMAIN ---
Rem Checks mailbox and reads letters.  Saved files are always directed to the
Rem main mail directory specified below.  Please customize the next two lines.
Rem
Mdir$="c:\mail\"                ! Main upper level mail directory.
Mysite$="logajan\"              ! Name of site mailbox directory.
```

```
Rem
Mwrk$=Mdir$+Mysite$+"WORK"
Md$=Mdir$+Mysite$+"*.*"
If Exist(Mwrk$)
  Kill Mwrk$
Endif
If Exist(Md$)
  Dir Md$ To Mwrk$
  Open "I",#3,Mwrk$
  M=0
  Bored!=False
  While Not Eof(#3)
    Line Input #3,Mfil$
    Mfix$=Mdir$+Mysite$+Mfil$
    If Exist(Mfix$)
      If Right$(Mfil$,2)=".X"
        Kill Mfix$
      Else
        If Mfil$<>"WORK"
          Inc M
          @More
          Repeat
            Cdn!=False
            Line Input "Next/Command";R$
            R$=Upper$(R$)
            If Len(R$)=0 Or Left$(R$,1)="N"
              Cdn!=True
            Endif
            If Left$(R$,1)="A" Or Left$(R$,1)="S"
              Snam$=Mid$(R$,3)
              If Len(Snam$)>0 And Len(Snam$)<13
                Open "I",#4,Mfix$
                If Left$(R$,1)="A"
                  If Exist(Mdir$+Snam$)
                    Open "A",#1,Mdir$+Snam$
                  Else
                    Print "File ";Mdir$;Snam$;" doesn't exist for appending";
                    Print " -- Creating it."
                    R$="S"
                    Open "O",#1,Mdir$+Snam$
                  Endif
                Else
                  Open "O",#1,Mdir$+Snam$
                Endif
                Lc=0
                While Not Eof(#4)
                  Inc Lc
                  Line Input #4,I$
                  Print #1,I$
                Wend
                Close #4
                Close #1
                If Left$(R$,1)="A"
```

```
                Print "File of ";Lc;" lines appended to ";Mdir$;Snam$
              Else
                Print "File of ";Lc;" lines saved in ";Mdir$;Snam$
              Endif
            Else
              Print "Filename - ";Snam$;" - too long."
            Endif
          Endif
          If Left$(R$,1)="D"
            Kill Mfix$
            Print "Message ";M;" deleted."
          Endif
          If Left$(R$,1)="T"
            If Exist(Mfix$)
              @More
            Else
              Print "Sorry, you have already deleted that message."
            Endif
          Endif
          If Left$(R$,1)="Q"
            Bored!=True
          Endif
          Exit If Bored!
          If Left$(R$,1)="H"
            @Hlp
          Endif
        Until Cdn!
      Endif
    Endif
  Endif
  Exit If Bored!
Wend
Close #3
Kill Mwrk$
If Not Bored!
  Print
  Print "No other mail."
Endif
Else
  Print "No mail."
Endif
Pause 50
End
Procedure More
  Open "I",#4,Mfix$
  Fll%=Lof(#4)
  Llt%=0
  Dne!=False
  Print Chr$(27);"EMessage - ";M
  Llc%=23
  Do
    For Lc=1 To Llc%
      Line Input #4,C$
```

```
      Print C$
      Llt%=Llt%+Len(C$)
      If Eof(#4)
        Dne!=True
      Endif
      Exit If Dne!
    Next Lc
    Exit If Dne!
    Pc%=Int(Llt%/Fll%*100)
    Pmt:
    Print Chr$(27);"p--MORE--";Chr$(27)+"q(";Pc%;"%)";
    R%=Inp(2)
    Print Chr$(27);"l";
    If R%=Asc("Q") Or R%=Asc("q")
      Dne!=True
    Endif
    Exit If Dne!
    If R%=&HD
      Llc%=1
    Else
      Llc%=23
    Endif
    If R%=Asc("T") Or R%=Asc("t")
      Close #4
      Open "I",#4,Mfix$
      Fll%=Lof(#4)
      Llt%=0
      Print Chr$(27);"EMessage - ";M
      Llc%=23
    Endif
    If R%=Asc("H") Or R%=Asc("h")
      @Hlp
      Goto Pmt
    Endif
  Loop
  Close #4
Return
Procedure Hlp
  Print
  Print "During message:"
  Print "   q - skips to end."
  Print "   t - reread from the top."
  Print "   CR - read next line."
  Print "   SPACE - read next page."
  Print "   h - print this help listing."
  Print
  Print "At command prompt:"
  Print "   n or CR - read next message."
  Print "   d - delete current message."
  Print "       (Undeleted messages remain in mailbox.)"
  Print "   t - reread current message."
  Print "   s filename - save message to file."
  Print "   a filename - append message to file."
```

```
  Print "   h - print this help listing."
  Print "   q - quit back to desktop."
  Print
Return
SHAR_EOF
cat << \SHAR_EOF > protocol.doc
                  Packet Driver Protocol

                       G. L. Chesson
                     Bell Laboratories
                (Shortened by John Logajan)
General

                            The technique depends on the
notion of a transmission window to determine upper and lower
bounds for valid sequence numbers.  The transmitter is
allowed to retransmit packets having sequence numbers within
the window until the receiver indicates that packets have
been correctly received.  Positive acknowledgement from the
receiver moves the window; negative acknowledgement or no
acknowledgement causes retransmission.  The receiver must
ignore duplicate transmission, detect the various errors
that may occur, and inform the transmitter when packets are
correctly or incorrectly received.

Packet Formats

     The protocol is defined in terms of message transmis-
sions of 8-bit bytes.  Each message includes one control
byte plus a data segment of zero or more information bytes.
The allowed data segment sizes range between 32 and 4096 as
determined by the formula 32(2k) where k is a 3-bit number.
The packet sequence numbers are likewise constrained to 3-
bits; i.e. counting proceeds modulo-8.

     The control byte is partitioned into three fields as
depicted below.

bit     7      6      5      4      3      2      1      0
        t      t      x      x      x      y      y      y

The t bits indicate a packet type and determine the
interpretation to be placed on the xxx and yyy fields.  The
various interpretations are as follows:

          tt      interpretation

          00      control packet
          10      data packet
          11      short data packet

A data segment accompanies all non-control packets.  Each
transmitter is constrained to observe the maximum data seg-
```

ment size established during initial synchronization by the receiver that it sends to. Type 10 packets have maximal size data segments. Type 11, or short, packets have zero or more data bytes but less than the maximum. The first one or two bytes of the data segment of a short packet are count bytes that indicate the difference between the maximum size and the number of bytes in the short segment. If the difference is less than 127, one count byte is used. If the difference exceeds 127, then the low-order seven bits of the difference are put in the first data byte and the high-order bit is set as an indicator that the remaining bits of the difference are in the second byte.

The sequence number of a non-control packet is given by the xxx field. Control packets are not sequenced. The newest sequence number, excluding duplicate transmissions, accepted by a receiver is placed in the yyy field of non-control packets sent to the other receiver.

There are no data bytes associated with a control packet, the xxx field is interpreted as a control message, and the yyy field is a value accompanying the control message. The control messages are listed below in decreasing priority. That is, if several control messages are to be sent, the lower-numbered ones are sent first.

xxx	name	yyy
1	CLOSE	n/a
2	RJ	last correctly received sequence number
3	---	---
4	RR	last correctly received sequence number
5	INITC	window size
6	INITB	data segment size
7	INITA	window size

The CLOSE message indicates that the communications channel is to be shut down. The RJ, or reject, message indicates that the receiver has detected an error and the sender should retransmit after using the yyy field to update the window. The RR, or receiver ready, message indicates that the receiver has detected no errors; the yyy field updates the sender's window. The INITA/B/C messages are used to set window and data segment sizes. Segment sizes are calculated by the formula 32(2yyy) and window sizes may range between 1 and 7.

Measurements of the protocol running on communication links at rates up to 9600 baud showed that a window size of 2 is optimal given a packet size greater than 32 bytes. This means that the link bandwidth can be fully utilized by the software.

Message Exchanges

Initialization

Messages are exchanged between four cooperating entities: two senders and two receivers. This means that the communication channel is thought of as two independent half-duplex data paths. For example the window and segment sizes need not be the same in each direction.

Initial synchronization is accomplished with two 3-way handshakes: two each of INITA/INITB/INITC. Each sender transmits INITA messages repeatedly. When an INITA message is received, INITB is sent in return. When an INITB message is received and an INITB message has been sent, an INITC message is sent. The INITA and INITB messages carry with them the packet and window size that each receiver wants to use, and the senders are supposed to comply. When a receiver has seen all three INIT messages, the channel is considered to be open. INIT messages are ignored elsewhere.

Data Transport

After initial synchronization each receiver sets a modulo-8 incrementing counter R to 0; each sender sets a similar counter S to 1. The value of R is always the number of the most recent correctly received packet. The value of S is always the first sequence number in the output window. Let W denote window size. Note that the value of W may be different for each sender.

A sender may transmit packets with sequence numbers in the range S to (S+W-1) mod-8. At any particular time a receiver expects arriving packets to have numbers in the range (R+1) mod-8 to (R+W) mod-8. Packets must arrive in sequence number order are are only acknowledged in order. That is, the next packet a receiver will acknowledge must have sequence number (R+1) mod-8.

A receiver acknowledges receipt of data packets by arranging for the value of its R counter to be sent across the channel where it will be used to update an S counter. This is done in two ways. If data is flowing in both directions across a channel then each receiver's current R value is carried in the yyy field of non-control packets. Otherwise when there is no bidirectional data flow, each receiver's R value is transmitted across the link as the yyy field of an RR control packet.

Error handling is up to the discretion of the receiver. It can ignore all errors in which case transmitter timeouts must provide for retransmission. The receiver may also generate RJ error control packets. The yyy field of an incom-

ing RJ message replaces the S value of the local sender and constitutes a request for retransmission to start at that sequence number.

Termination

The CLOSE message is used to terminate communications. Software on either or both ends of the communication channel may initiate termination. In any case when one end wants to terminate it sends CLOSE messages until one is received from the other end or until a programmable limit on the number of CLOSE messages is reached. Receipt of a CLOSE message causes a CLOSE message to be sent.

Framing

The term framing is used to denote the technique by which the beginning and end of a message is detected in a byte stream; error control denotes the method by which transmission errors are detected.

A six byte framing envelope is constructed using the control byte C of a packet and five other bytes as depicted below.

<DLE><k><c0><c1><C><x>

The <DLE> symbol denotes the ASCII ctrl/P character. If the envelope is to be followed by a data segment, <k> has the value log2(size)-4; i.e. 1 < k < 8. If k is 9, then the envelope represents a control packet. The <c0> and <c1> bytes are the low-order and high-order bytes respectively of a 16-bit checksum. For control packets, this value is 0xAAAA minus the control byte C.
For data packets, the checksum is calculated by a program. (John Logajan's 68000 version is below.)
The <x> byte is the exclusive-or of <k><c0><c1><C>. Error control is accomplished by checking a received framing envelope for compliance with the definition, and comparing a checksum function of the data segment with <c0><c1>.

This particular framing strategy assumes data segments are constant-sized: the unused bytes in a short packet are actually transmitted.

Logajan's 68000 version of the checksum program,
as called from a GFA Basic C: call.

```
start: move.l 4(sp),a0      ;data packet start address
       move.w 8(sp),d1      ;data packet length
       move.w 10(sp),d2     ;command byte
       move.l #$ffff,d0
       clr.w d3
ck1:   rol.w #1,d0
```

```
       move.l d0,d4
       clr.l d5
       move.b (a0)+,d5
       add.w d5,d0
       move.w d0,d5
       eor.w d1,d5
       add.w d5,d3
       cmp.l d4,d0
       bgt.s ovck
       eor.w d3,d0
ovck:  sub.w #1,d1
       bne.s ck1
       and.w #$ffff,d2
       eor.w d0,d2
       move.w #$aaaa,d0
       sub.w d2,d0
       and.l #$ffff,d0      ;Checksum in D0.
       rts
```

Example session:

Note below: S = we send, R = we receive.
After the login: and password: stuff;

```
R Shere[=sitename]
S Soursitename
R ROK
R Pg[other protcols]
S Ug
S INITA (with our window size)
R INITA (with their window size)
S INITB (with our packet size)
R INITB (with their packet size)
S INITC (with our window size, again)
R INITC
S S workline with D.file name.   (If we have work, else H)
R RR
R SY
S RR
S data packets, short packet if need, followed by empty pkt.
R RR's
R CY
S RR
S S workline with X.file name.
R RR
R SY
S data packets
R RR's
R CY
S H    (If we are done)
R RR
R HN   (If they have work, else HY)
S RR
```

```
R S workline D.file
S RR
S SY
R RR
R data packets
S RR's
S CY
R RR
R S workline X.file
S RR
S SY
R RR
R data packets
S RR's
S CY
R RR
R H   (If they are done too.)
S RR
S HY
R RR
R HY
R RR
S CLOSE
R CLOSE
S OOOOOO
R OOOOOO (Hangup)
-----------------------------------------------------

SHAR_EOF
#       End of shell archive
exit 0


From -3361494846976768533
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,d89cc069f1112b22,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-18 10:40:44 PST
Path: utzoo!attcan!uunet!wyse!vsi1!ames!pasteur!ucbvax!agate!saturn!ssyx.ucsc.edu!
koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i029: 1stjet -- 1st-Word configuration file for DeskJet
Keywords: raw
Message-ID: <6722@saturn.ucsc.edu>
Date: 18 Mar 89 18:40:44 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 210
Approved: koreth@ssyx.ucsc.edu

Submitted-by: sdsu.sdsu.edu!ucselx!coffey (Pat Coffey)
Posting-number: Volume 2, Issue 29
Archive-name: 1stjet
```

To switch from draft to letter quality on the desk jet. You should
rename the file deskjet.hex, edit off this part, and run install.prg
to produce the 1st_print.dot file.

```
---------------------cut here-------------------------
*****************************************************************
*
*    Hewlett-Packard DeskJet Printer Driver Configuration Table
*    For use with First Word (public domain version)
*    This file contains tables defining the code sequences
*    to be sent to the printer to perform various functions
*    and to access the characters from codes in the Atari
*    character set.
*    Author: Frederic MORA
*            Pat Coffey
*    Modification history
*          PaC 1988     Download Fredric MORA's first word plus driver
*                       confifguration table from GENIE
*          PaC 9/25/88  Change italic from double underline to compressed
*                       add lq & draft codes
*****************************************************************

*
* Name of printer
* ===============
*
HP DeskJet


*    Configuration variables
*    =======================
*
*    These six variables define certain printer
*    configuration details required by 1st Word Plus:
*
*    (1)  1 for microspacing daisywheel, else 0
*    (2)  Character width in printer units (daisy)
*    (3)  Linefeed height in printer units
*    (4)  Mid-carriage (daisy) pixels/inch (matrix) (96H for 150 dpi)
*    (5)  Bold offset in printer units (daisy)
*    (6)  1 to pause printing between pages, else 0


0, 0, C, 96, 0, 0


*
* Printer characteristics
* =======================
*
* This table specifies the printer command sequences.
* If the top bit of a code is set, then this indicates the position
```

```
* of a parameter passed to the printer. The code whose top bit is set
* in this table is added to the parameter passed before being sent to the
* printer. It is not used in all command sequences, only in those where
* the printer requires  a variable value such as the length of a vertical
* tab.
*
* 0            * Character width
1, D, A        * Linefeed WITH return
* 2            * Forward print
* 3            * Reverse print
* 4            * Vertical tab to line
* 5            * Absolute horizontal tab
6, 1B, 28, 73, 31, 71, 33, 42      * Draft bold on:  Esc ( s  2 q 3 B
7, 1B, 28, 73, 31, 71, 30, 42      * Draft bold off Esc ( s 2 q 0 B
8, 1B, 28, 73, 32, 71, 33, 42    * Near Letter Quality bold on:Esc ( s 3 B
9, 1B, 28, 73, 32, 71 30, 42      * NLQ bold off:             Esc ( s 0 B
* "Draft italic" on (compressed):   Esc ( s 1 q  6 V
A, 1B, 28, 73, 31, 71, 36, 56
B, 1B, 28, 73, 31, 71, 31, 30, 56  * Draft italic off   esc ( s 1 q 10 h 12 V
* "NLQ italic" on (compressed)      Esc ( s 2 q 10 h 12 V
C, 1B, 28, 73, 32, 71, 36, 56      * NLQ italic on
D, 1B, 28, 73, 32, 71, 31, 30, 56     * NLQ italic off            Esc & d @
E, 1B, 28, 73, 31, 71, 31, 36, 68, 36, 56       * Draft light on (6 point font)
F,  1B, 28, 73, 31, 71,31, 30, 68, 31, 30, 56             * Draft light off
10, 1B, 28, 73, 32, 71, 31, 36, 68, 36, 56   * NLQ light on   (6 point font)
11, 1B, 28, 73, 32, 71, 31, 30, 68, 31, 30, 56   * NLQ light off
* Draft superscript on with 6 point superscript: Esc ( s 6 V Esc ( s +1 U
12, 1B, 28, 73, 31, 71, 36, 56, 1B, 28, 73, 2B, 31, 55
 * Draft superscript off (back to 12 point):     Esc ( s 0 U Esc ( s 12 V
13, 1B, 28, 73, 31, 71, 30, 55, 1B, 28, 73, 31, 32, 56
* NLQ superscript on with 6 point superscript:   Esc ( s 6 V Esc ( s +1 U
14, 1B, 28, 73, 32, 71, 36, 56, 1B, 28, 73, 2B, 31, 55
* NLQ superscript off (back to 12 point):        Esc ( s 0 U Esc ( s 12 V
15, 1B, 28, 73, 32, 71, 30, 55, 1B, 28, 73, 31, 32, 56
* Draft subscript on with 6 point subscript:     Esc ( s -1 U Esc ( s 6 V
16, 1B, 28, 73, 31, 71, 2D, 31, 55, 1B, 28, 73, 36, 56
* Draft subscript off (back to 12 point):        Esc ( s 0 U Esc ( s 12 V
17, 1B, 28, 73, 31, 71, 30, 55, 1B, 28, 73, 31, 32, 56
* NLQ subscript on with 6 point subscript:       Esc ( s -1 U Esc ( s 6 V
18, 1B, 28, 73, 32, 71, 2D, 31, 55, 1B, 28, 73, 36, 56
* NLQ subscript off (back to 12 point):          Esc ( s 0 U Esc ( s 12 V
19, 1B, 28, 73, 32, 71, 30, 55, 1B, 28, 73, 31, 32, 56
* Draft underline (single fixed) on:             Esc & d 1 D
1A, 1B, 26, 64, 31, 44
1B, 1B, 26, 64, 40  * Draft underline off:       Esc & d @
* NLQ underline (single fixed) on:               Esc & d 1 D
1C, 1B, 26, 64, 31, 44
1D, 1B, 26, 64, 40  * NLQ underline off:         Esc & d @
1E, 1B, 26, 6C, 30, 48   * Formfeed              Esc & l 0 H
*1F, 12         * Horizontal initialisation
*20, C          * Vertical initialisation
21, 1B, 45      * Termination: printer reset     Esc E
22,  8                       * Backspace
```

```
23,  D                        * Carriage return
*24, 1B, 43, 80                * Form length in lines
*25, 1B, 41, 80, 1B, 32        * Set line feed to N units
*26  * Start graphics
* 27                           * Reserved

0              * NULL termination byte

*
* Translation Table
* =================
*
* This table provides translation from single Atari input bytes into
* multiple printer codes, and is useful for printing extraneous
* characters such as accented characters etc. All characters are
* subjected to translation, but if there is no entry in the table for
* a particular code, then the original code is sent to the printer.
*
* The entries must be arranged in ascending order of Atari input
* code. The table is NULL terminated.
*
* This table assumes that the HP DeskJet printer is set in
* PC8 Character Set (Switches A1 to A4 down)

0                                  * NULL: print a space
1             * No Up arrow
2             * No Down arrow
3             * No Right arrow
4             * No Left arrow
5             * No close box
6             * No size box
7             * No full box
8             * No tick
9             * No clock
A             * No bell
B, 0D                              * musical note
E             * No LH Atari symbol
F             * No RH Atari symbol
10, 30        * LCD 0
11, 31        * LCD 1
12, 32        * LCD 2
13, 33        * LCD 3
14, 34        * LCD 4
15, 35        * LCD 5
16, 36        * LCD 6
17, 37        * LCD 7
18, 38        * LCD 8
19, 39        * LCD 9
7F                        * No triangle
9E, E1                    * Esszet
B0         * No Lower case a tilde
B1         * No Lower case o tilde
B2         * No Capital crossed O
```

```
B3         * No Lower case crossed o
B4         * No lower case oe dipthong
B5                       * No capital OE dipthong
B6, 41                   * No capital A grave: print A
B7, 41                   * No capital A tilde: print A
B8, 4F                   * No capital O tilde: print O
B9         * No Umlaut
BA, 27                   * Acute: print '
BB, C5                   * Dagger
BC, 15                   * Paragraph symbol
BD                       * No copyright symbol
BE                       * No Registered symbol
BF                       * No Trademark symbol
C0        * No ij ligature
C1        * No IJ ligature
C2        * No Hebrew...
C3
C4
C5
C6
C7
C8
C9
CA
CB
CC
CD
CE
CF
D0
D1
D2
D3
D4
D5
D6
D7
D8
D9
DA
DB
DC
DD, 14       * Section mark
DE, 5E       * Dropped circumflex
DF, EC       *  infinity
FE           * No superior 3
FF        * No macron
0


From 4959731041521639841
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,1666e8ae3f0f66c6,start
```

X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-18 10:44:43 PST
Path: utzoo!attcan!uunet!wyse!vsi1!ames!pasteur!ucbvax!agate!saturn!ssyx.ucsc.edu!
koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i030: 1stjethv -- 1st-Word config file for Helvetica on DeskJet
Keywords: raw
Message-ID: <6723@saturn.ucsc.edu>
Date: 18 Mar 89 18:44:43 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 258
Approved: koreth@ssyx.ucsc.edu

Submitted-by: sdsu.sdsu.edu!ucselx!coffey (Pat Coffey)
Posting-number: Volume 2, Issue 30
Archive-name: 1stjethv

From Genie and modified to use the Helvitica font cartridge.
You should edit off the mail info, and rename the file djhelvit.hex, and
run install.prg to get the 1st_print.dot file you want.

```
----------------------------cut here--------------------------------
*****************************************************************
*
*    Hewlett-Packard DeskJet Printer Driver Configuration Table
*    For use with First Word Plus
*    This file contains tables defining the code sequences
*    to be sent to the printer to perform various functions
*    and to access the characters from codes in the Atari
*    character set.
*    Author: Frederic MORA
*            Pat Coffey
*    Modification history
*          PaC 1988     Download Fredric Mora's version from GENIE
*          PaC 9/25/88  Change italic from double underline to compressed
*                       add lq & draft codes
*          PaC 12/25/88 add helvitica fonts
*
*****************************************************************

*
* Name of printer
* ===============
*
HP DeskJet with Helvitica 22706Q cartridge


*    Configuration variables
*    =======================
*
*    These six variables define certain printer
*    configuration details required by 1st Word Plus:
```

```
*
*    (1)  1 for microspacing daisywheel, else 0
*    (2)  Character width in printer units (daisy)
*    (3)  Linefeed height in printer units
*    (4)  Mid-carriage (daisy) pixels/inch (matrix) (96H for 150 dpi)
*    (5)  Bold offset in printer units (daisy)
*    (6)  1 to pause printing between pages, else 0


0, 0, C, 96, 0, 0


*
* Printer characteristics
* ======================
*
* This table specifies the printer command sequences.
* If the top bit of a code is set, then this indicates the position
* of a parameter passed to the printer. The code whose top bit is set
* in this table is added to the parameter passed before being sent to the
* printer. It is not used in all command sequences, only in those where
* the printer requires  a variable value such as the length of a vertical
* tab.
*
* 0            * Character width
1, D, A        * Linefeed WITH return
* 2            * Forward print
* 3            * Reverse print
* 4            * Vertical tab to line
* 5            * Absolute horizontal tab
*           6, * Draft bold on: Esc(0U Esc ( s  2 q 3 b 4 T
* Esc ( 0  U  Esc (  s  0  u  1  p  1  4  v  3  b  4  t  1  Q
6,1B,28,30,55,1B,28,73,30,75,31,70,31,32,76,33,62,34,74,31,51            * No Up
arrow INIT HELV. 12 PT.

* 7         * Draft bold off
* Esc ( 0  U  Esc (  s  0  u  1  p  1  4  v  0  b  4  t  1  Q
7,1B,28,30,55,1B,28,73,30,75,31,70,31,32,76,30,62,34,74,31,51            * No Up
arrow INIT HELV. 12 PT.

* 8,        * Near Letter Quality bold on:
* Esc ( 0  U  Esc (  s  0  u  1  p   1  4  v  3  b  4  t  2  Q
8,1B,28,30,55,1B,28,73,30,75,31,70,31,32,76,,33,62,34,74,32,51            * No Up
arrow INIT HELV. 12 PT.

* 9,        * NLQ bold off:
* Esc ( 0  U  Esc (  s  0  u  1  p  1  4  v  0  b  4  t  2  Q
9,1B,28,30,55,1B,28,73,30,75,31,70,31,32,76,30,62,34,74,32,51            * No Up
arrow INIT HELV. 12 PT.

* A,        Draft italic
* Esc ( 0  U  Esc (  s  0  u  1  p  1  4  v  1  s  4  t  1  Q
A,1B,28,30,55,1B,28,73,30,75,31,70,31,32,76,31,73,34,74,31,51            * No Up
```

```
arrow INIT HELV. 12 PT.
* B,        * Draft italic off
* Esc ( 0  U  Esc (  s  0  u  1  p  1  4  v  0  s  4  t  1  Q
B,1B,28,30,55,1B,28,73,30,75,31,70,31,32,76,30,73,34,74,31,51            * No Up
arrow INIT HELV. 12 PT.

* C         NLQ italic on
* Esc ( 0  U  Esc (  s  0  u  1  p  1  4  v  1  s  4  t  2  Q
C,1B,28,30,55,1B,28,73,30,75,31,70,31,32,76,31,73,34,74,32,51            * No Up
arrow INIT HELV. 12 PT.


* D,         * NLQ italic off
* Esc ( 0  U  Esc (  s  0  u  1  p  1  4  v  0  s  0  b  4  t  2  Q
D,1B,28,30,55,1B,28,73,30,75,31,70,31,32,76,30,73,30,62,34,74,32,51            *
No Up arrow INIT HELV. 12 PT.

* E          * Draft light on (8 point font)
* Esc ( 0  U  Esc (  s  0  u  1  p  8  v  0  s  0  b  4  t  1  Q
E,1B,28,30,55,1B,28,73,30,75,31,70,38,76,30,73,30,62,34,74,31,51            * No
Up arrow INIT HELV. 12 PT.

* F,         * Draft light off
* Esc ( 0  U  Esc (  s  0  u  1  p  1  2  v  0  b  4  t  1  Q
F,1B,28,30,55,1B,28,73,30,75,31,70,31,32,76,30,62,34,74,31,51            * No Up
arrow INIT HELV. 12 PT.

* 10         * NLQ light on   (8 point font)
*  Esc ( 0  U  Esc (  s  0  u  1  p  8  v  0  b  4  t  2  Q
10,1B,28,30,55,1B,28,73,30,75,31,70,38,76,30,62,34,74,32,51            * No Up
arrow INIT HELV. 12 PT.

* 11         * NLQ light off   (BACK TO 12 POINT)
* Esc ( 0  U  Esc (  s  0  u  1  p  1  4  v  0  b  4  t  2  Q
11,1B,28,30,55,1B,28,73,30,75,31,70,31,32,76,30,62,34,74,32,51            * No Up
arrow INIT HELV. 12 PT.

* Draft superscript on with 8 point superscript: Esc ( s 8 V Esc ( s +1 U
12,1B,28,73,31,71,38,56,1B,28,73,2B,31,55
 * Draft superscript off (back to 12 point):     Esc ( s 0 U Esc ( s 12 V
13,1B,28,73,31,71,30,55,1B,28,73,31,32,56
* NLQ superscript on with 8 point superscript:   Esc ( s 8 V Esc ( s +1 U
14,1B,28,73,32,71,38,56,1B,28,73,2B,31,55
* NLQ superscript off (back to 12 point):        Esc ( s 0 U Esc ( s 12 V
15,1B,28,73,32,71,30,55,1B,28,73,31,32,56
* Draft subscript on with 8 point subscript:     Esc ( s -1 U Esc ( s 8 V
16,1B,28,73,31,71,2D,31,55,1B,28,73,38,56
* Draft subscript off (back to 12 point):        Esc ( s 0 U Esc ( s 12 V
17,1B,28,73,31,71,30,55,1B,28,73,31,32,56
* NLQ subscript on with 8 point subscript:       Esc ( s -1 U Esc ( s 8 V
18,1B,28,73,32,71,2D,31,55,1B,28,73,38,56
* NLQ subscript off (back to 12 point):          Esc ( s 0 U Esc ( s 12 V
19,1B,28,73,32,71,30,55,1B,28,73,31,32,56
```

```
* Draft underline (single fixed) on:             Esc & d 1 D
1A,1B,26,64,31,44
1B,1B,26,64,40  * Draft underline off:       Esc & d @
* NLQ underline (single fixed) on:              Esc & d 1 D
1C,1B,26,64,31,44
1D,1B,26,64,40  * NLQ underline off:         Esc & d @
1E,1B,26,6C,30,48   * Formfeed              Esc & l 0 H
*1F,12         * Horizontal initialisation
20, 1B,26,6B,25,57          * Vertical initialisation
21,1B,45      * Termination: printer reset     Esc E
22, 8                        * Backspace
23, D                        * Carriage return
*24,1B,43,80                * Form length in lines
*25,1B,41,80,1B,32        * Set line feed to N units
*26  * Start graphics
* 27                          * Reserved

0              * NULL termination byte

*
* Translation Table
* =================
*
* This table provides translation from single Atari input bytes into
* multiple printer codes,and is useful for printing extraneous
* characters such as accented characters etc. All characters are
* subjected to translation,but if there is no entry in the table for
* a particular code,then the original code is sent to the printer.
*
* The entries must be arranged in ascending order of Atari input
* code. The table is NULL terminated.
*
* This table assumes that the HP DeskJet printer is set in
* PC8 Character Set (Switches A1 to A4 down)

0                                * NULL: print a space
* the up arrow code will set the printer for 14 pt. Helvitica
* Esc ( 0  U  Esc (  s  1  p  1  4  v  4  T
1,1B,28,30,55,1B,28,73,31,70,31,34,76,34,54            * No Up arrow INIT HELV.
14 PT.
* 2             * No Down arrow so fake helvitica 10 point
* Esc ( 0  U  Esc (  s  1  p  1  0  v  4  T
2,1B,28,30,55,1B,28,73,31,70,31,30,76,34,54            * No Up arrow INIT HELV.
14 PT.
* 3             * No Right arrow Fake helvitica 7 point
* Esc ( 0  U  Esc (  s  1  p  7 v  4  T
3,1B,28,30,55,1B,28,73,31,70,37,76,34,54            * No Up arrow INIT HELV. 14
PT.
* 4             * No Left arrow so fake 6 point
* Esc ( 0  U  Esc (  s  1  p  6  v  4  T
4,1B,28,30,55,1B,28,73,31,70,36,76,34,54            * No Up arrow INIT HELV. 14
PT.
* 5             * No close box so fake 5 point
```

```
* Esc ( 0  U  Esc (  s  1  p  5  v  4  T
5,1B,28,30,55,1B,28,73,31,70,35,76,34,54            * No Up arrow INIT HELV. 14
PT.
* 6             * No size box so fake 4 point
* Esc ( 0  U  Esc (  s  1  p  4  v  4  T
6,1B,28,30,55,1B,28,73,31,70,34,76,34,54            * No Up arrow INIT HELV. 14
PT.

7             * No full box
8             * No tick
9             * No clock
A             * No bell
B,0D                              * musical note
E             * No LH Atari symbol
F             * No RH Atari symbol
10,30        * LCD 0
11,31        * LCD 1
12,32        * LCD 2
13,33        * LCD 3
14,34        * LCD 4
15,35        * LCD 5
16,36        * LCD 6
17,37        * LCD 7
18,38        * LCD 8
19,39        * LCD 9
7F                       * No triangle
9E,E1                   * Esszet
B0         * No Lower case a tilde
B1         * No Lower case o tilde
B2         * No Capital crossed O
B3         * No Lower case crossed o
B4         * No lower case oe dipthong
B5                        * No capital OE dipthong
B6,41                   * No capital A grave: print A
B7,41                   * No capital A tilde: print A
B8,4F                   * No capital O tilde: print O
B9         * No Umlaut
BA,27                   * Acute: print '
BB,C5                   * Dagger
BC,15                   * Paragraph symbol
BD                       * No copyright symbol
BE                       * No Registered symbol
BF                       * No Trademark symbol
C0        * No ij ligature
C1        * No IJ ligature
C2        * No Hebrew...
C3
C4
C5
C6
C7
C8
C9
```

```
CA
CB
CC
CD
CE
CF
D0
D1
D2
D3
D4
D5
D6
D7
D8
D9
DA
DB
DC
DD,14       * Section mark
DE,5E       * Dropped circumflex
DF,EC       *  infinity
FE           * No superior 3
FF        * No macron
0


From 8222214828223123319
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,cfe383b6a051190d,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-18 10:57:17 PST
Path: utzoo!attcan!uunet!wyse!vsi1!ames!pasteur!ucbvax!agate!saturn!ssyx.ucsc.edu!
koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i031: cpboot -- Copy executable boot sectors between floppy formats
Keywords: shar, C
Message-ID: <6724@saturn.ucsc.edu>
Date: 18 Mar 89 18:57:17 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 211
Approved: koreth@ssyx.ucsc.edu

Submitted-by: math.waterloo.edu!orchid!achowe (Anthony C. Howe?)
Posting-number: Volume 2, Issue 31
Archive-name: cpboot

#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      README
```

```
#      CPBOOT.C
#
sed 's/^X//' << 'SHAR_EOF' > README &&
Xreadme  881015
X
X
XCPBOOT.TOS
X==========
X
Xobjective
X---------
XThis program is designed to copy an executable boot sector from a disk of
Xone type of format to that of an other.
X
Xwhy
X---
XThe reason for this is that I wanted to make copies of TOS 1.4 Developer's
XRelease and place them on double sided disks. Well TOS 1.4 comes on a single
Xsided disk and Atari did not provided a program to modify a boot sector with
Xthe TOS loader. The only possible way to make copies was with GEM's disk to
Xdisk copy, which ment that I could only make single sided copies. I then tried
Xusing MEMFIL14.ACC to read the boot sector from the TOS disk to that of a
Xdouble sided disk. ** Smooth move Ex-lax **!!  I had forgotten that certain
Xbits of info like number of tracks and sides, and the serial number, would be
Xdifferent on the two disks. So I ended up trashing a disk and all the info on
Xit.
X
Xwhat
X----
XBasically all CPBOOT does is to read both the source and target boot sectors
Xfrom respective disks, copies from the source the branch code, filler, and
Xthe boot sector code to the target sector. Then a call to Protobt() is made
Xin order to adjust the checksum for an executable boot sector, which is then
Xwritten back to the target disk. See CPBOOT.C.
X
Xhow
X---
XJust execute the program CPBOOT.TOS and follow the prompts. All work is done
Xon A drive so disk swapping will be necessary.
X
XDisclaimer
X----------
XI do not take responsibilty for damage caused by careless use of this program.
XMake sure that you follow the prompts, write tabs are up on disks you want to
Xprotect and that you have backups of everything. This program was a one day
Xquick utility. You can modify the source for more checks like: does the target
Xalready have a boot sector, have the disks actually been swapped, prompts
Xto select A: or B:, etc.
X
XEnvironment
X-----------
XMega ST2 with colour screen.
XBDT's GEMCSH & MAKE
```

```
XOld Acylon C
XALN 88/08/12 Linker
XMADMAC v1.00
XACHAPP.S and ACHSLIB.S (Anthony's assembler library v1.02)
X
X
XHope this utility comes in handy.
X
X- Ant
SHAR_EOF
chmod 0600 README || echo "restore of README fails"
sed 's/^X//' << 'SHAR_EOF' > CPBOOT.C &&
X/*
X    cpboot.c      Copy Boot Sector
X
X    881014    ACH
X
X*/
X
X#include <osbind.h>
X
Xextern unsigned strlen();
X
X#define SECTOR             512
X#define DELAY_FACTOR  3276
X#define SCREEN_WIDTH  80
X#define SCREEN_LENGTH 25
X#define CURSOR_UP     "\033A"
X#define CURSOR_DOWN        "\033B"
X#define CURSOR_RIGHT  "\033C"
X#define CURSOR_LEFT        "\033D"
X#define CURSOR_HOME        "\033H"
X#define CURSOR_POSITION "\033Y"
X#define CURSOR_OFFSET 32
X#define REVERSE_INDEX "\033I"
X#define CLEAR_SCREEN  "\033E"
X#define CLEAR_EOP     "\033J"
X#define CLEAR_EOL     "\033K"
X#define LINE_INSERT        "\033L"
X#define LINE_DELETE        "\033M"
X
X
Xtypedef int void;
X
Xint ScreenWidth = SCREEN_WIDTH -1;
Xint ScreenLength = SCREEN_LENGTH -1;
Xchar source[ SECTOR ];
Xchar target[ SECTOR ];
X
X
Xvoid RowCol( row, col )
X    int row;
X    int col;
```

```
X{
X    Cconws( CURSOR_POSITION );
X    Cconout( row + CURSOR_OFFSET );
X    Cconout( col + CURSOR_OFFSET );
X} /* RowCol */
X
X
Xvoid RowClear( row )
X    int row;
X{
X    RowCol( row, 0 );
X    Cconws( CLEAR_EOL );
X} /* RowClear */
X
X
Xvoid Dclear()
X{
X    Cconws( CLEAR_SCREEN );
X} /* Dclear */
X
X
Xvoid Scenter( row, str )
X    int row;
X    char* str;
X{
X    int x;
X
X    x = strlen( str );
X    RowClear( row );
X    RowCol( row, ScreenWidth /2 - x /2 );
X    Cconws( str );
X} /* Scenter */
X
X
Xvoid Wait()
X{
X    Scenter( ScreenLength, "[ press a key ]" );
X    while( !Cconis() );
X    Cconin();
X    RowClear( ScreenLength );
X} /* Wait */
X
X
Xvoid Delay( x )
X    int x;
X{
X    for( x *= DELAY_FACTOR; x; --x );
X} /* Delay */
X
X
Xvoid main()
X{
X    long serial;
```

```
X    short sides;
X    short tracks;
X
X    Dclear();
X    Scenter( 8, "Copy Boot Sector" );
X    Scenter( 9, "================" );
X
X    Scenter( 12, "Insert SOURCE disk into A:." );
X    Wait();
X
X    /* Read source sector. */
X    Scenter( 12, "Reading source boot sector." );
X    Floprd( source, 0L, 0, 1, 0, 0, 1 );
X
X    Scenter( 12, "Insert TARGET disk into A:." );
X    Wait();
X
X    /* Read target sector. */
X    Scenter( 12, "Reading target boot sector." );
X    Floprd( target, 0L, 0, 1, 0, 0, 1 );
X
X    Scenter( 12, "Merging source sector with target sector." );
X    Delay( 10 );
X
X    /* Copy branch and filler. */
X    memmove( target, source, 8L );
X    /* Copy boot code. */
X    memmove( &target[ 0x1e ], &source[ 0x1e ], 0x200L -0x1eL );
X    /* Prototype boot sector to readjust checksum. */
X    Protobt( target, -1L, -1, 1 );
X
X    /* Update target boot sector. */
X    Scenter( 12, "Writing updated target boot sector." );
X    Flopwr( target, 0L, 0, 1, 0, 0, 1 );
X
X    Scenter( 12, "Done." );
X    Wait();
X} /* main */
X
X
SHAR_EOF
chmod 0600 CPBOOT.C || echo "restore of CPBOOT.C fails"
exit 0


From 5542049600318480460
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,eb1b36eab062e7ee,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-19 13:58:26 PST
Path: utzoo!attcan!uunet!wyse!vsi1!ames!mailrus!tut.cis.ohio-state.edu!ucbvax!
agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
```

Newsgroups: comp.sources.atari.st
Subject: v02i032: du -- List disk usage in directory trees
Keywords: shar, C
Message-ID: <6730@saturn.ucsc.edu>
Date: 19 Mar 89 21:58:26 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 190
Approved: koreth@ssyx.ucsc.edu

Submitted-by: juancho@dgp.toronto.edu (John Buchanan)
Posting-number: Volume 2, Issue 32
Archive-name: du

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      DU.C
#      DU.DOC
#      MAKEFILE
#
sed 's/^X//' << 'SHAR_EOF' > DU.C &&
X/*
X *
X *
X *  Program: du
X *  Programmer:   John Buchanan
X *
X *  Print out the disc usage from the current directory.
X *       or directories in the arguments.
X *  -s means print out only the top level information.
X *
X *  Copyright:    Public domain code;
X *
X *
X */
X
X/*-----------------------------------------------Include files-----------*/
X
X#include <osbind.h>
X#include <stat.h>
X#include <ctype.h>
X
X/*-----------------------------------------------External variables------*/
X
Xextern  long _stksize;          /* to check stack size  */
X
X/*-----------------------------------------------Defines-----------------*/
X
X/* I defined this as a macro so that it would be easier to use in other
X * programs.
X */
X
```

```
X#define FALSE (1==0)
X#define TRUE  (1==1)
X
X#define STACKCHECK()  DMABUFFER  *saved;          \
X             if ((long)&saved <= _stksize+128)    \
X             {                         \
X             printf("Stack about to over flow\n\r");   \
X             exit(1);                  \
X             }
Xint SummaryOnly;
X
Xmain(argc,argv)
Xint argc;
Xchar *argv[];
X    {
X    char *Current=".";      /* Set the current dir */
X    int  i=1;
X    SummaryOnly = FALSE;
X    if(argc > 1)
X         if(argv[1][0] == '-' && (argv[1][1] == 's' ||
X                                  argv[1][1] == 'S') )
X             {
X             SummaryOnly = TRUE;
X             i++;
X             }
X    if(argc == i)
X         printf("%s %d\n\r",Current,DirTotal(Current));
X    else
X         for(;i< argc; i++)
X              printf("%s %d\n\r",argv[i],DirTotal(argv[i]));
X    exit(0);
X    }
X
Xchar *lowercase(name)
Xregister char *name;
X    {
X    register char *p = name;
X    register char c;
X    while (c = *p)
X         *p++ = (isascii(c) && isupper(c)) ? tolower(c) : c ;
X    return (name);
X    }
X
Xchar *dircat(pfx,sfx)
Xregister char *pfx, *sfx;
X/*
X *  Code from MWC Manual
X */
X    {
X    extern char *malloc(),
X             *strcat();
X    register char *p;
X    register int nb, npfx;
```

```
X    nb = (npfx = strlen(pfx)) + 1 + strlen(sfx) +1;
X    if ((p = malloc(nb)) == 0)
X         {
X         printf("Failed to allocate memory");
X         exit(1);
X         }
X    strcpy(p,pfx);
X    if (npfx !=0 && pfx[npfx-1]!='\\')
X         strcat(p,"\\");
X    return strcat(p,sfx);
X    }
Xint DirTotal(name)
Xchar    *name;
X    {
X    register char *globname, *newname;
X    register total=0;
X    DMABUFFER dumb;
X    STACKCHECK();               /* check the stack */
X    globname = dircat(name,"*.*");   /* search all files*/
X    saved=(DMABUFFER *) Fgetdta();       /* Save dta      */
X    Fsetdta(&dumb);                  /* Get dta         */
X
X    if (Fsfirst(globname,0xFF) == 0) /* there are some files   */
X         {
X         do
X              {
X              if (dumb.d_fattr == 0x10 && /* directory get total */
X                   dumb.d_fname[0] != '.' ) /* not current    */
X                   {
X                   int tmp;
X                   newname = dircat(name,dumb.d_fname);
X                   tmp = DirTotal(newname);
X                   if (!SummaryOnly)
X                        printf("%s %d \n\r",lowercase(newname),tmp);
X                   total += tmp;
X                   free(newname);
X                   }
X              else
X                   total += (dumb.d_fsize >> 10) + 1;
X                                 /* Divide by 1024  */
X              }
X         while (Fsnext() == 0);
X         }
X    free(globname);                      /* release mem
       */
X    Fsetdta(saved);                      /* restore dta
       */
X    return(total);                       /* return total of
X                                   current dir     */
X
X    }
X
SHAR_EOF
```

```
chmod 0600 DU.C || echo "restore of DU.C fails"
sed 's/^X//' << 'SHAR_EOF' > DU.DOC &&
X(L)          Gulam Commands(L)
X
XNAME:  du
X
XSYNOPSIS
X    du [-s] [directories]
X
XOPERATION
X    Recursively list the sizes of the specified directories (default is
X'.').  Output looks like this.
X
XAUTHOR
X    John Buchanan
X
XFILES
X    du.prg
X    du.doc
X
XBUGS
X    Probably
X
X
SHAR_EOF
chmod 0600 DU.DOC || echo "restore of DU.DOC fails"
sed 's/^X//' << 'SHAR_EOF' > MAKEFILE &&
XCFLAGS = -VPEEP -A
X
XBIN = e:\ebin
X
Xdu.prg: du.c
X    cc -o du.prg $(CFLAGS) du.c
Xinstall: du.prg
X    strip du.prg
X    cp du.prg $(BIN)
Xdist:
X    arc a du.arc du.c makefile du.prg du.doc
X    uuencode du.arc
SHAR_EOF
chmod 0600 MAKEFILE || echo "restore of MAKEFILE fails"
exit 0


From 7776567710721127599
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,7b8ad3432a515596,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-19 14:19:54 PST
Path: utzoo!attcan!uunet!wyse!vsi1!ames!mailrus!tut.cis.ohio-state.edu!ucbvax!
agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
```

Subject: v02i033: fplib10 -- Floating point routines for Sozobon C
Keywords: shar, C
Message-ID: <6732@saturn.ucsc.edu>
Date: 19 Mar 89 22:19:54 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 712
Approved: koreth@ssyx.ucsc.edu

Submitted-by: brooks@csss-a.prime.com (David Brooks)
Posting-number: Volume 2, Issue 33
Archive-name: fplib10

[These are the sources to some floating-point library routines for Sozobon C.
 Apparently this is not the complete floating point library; the whole thing,
 in binary-only form, has been posted to the binaries group. -sg]

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      ATOF.C
#      EXP.C
#      FABS.C
#      FMOD.C
#      FPCMP.S
#      ITRIG.C
#      LOG.C
#      MATH.H
#      POW.C
#      README
#      SQRT.C
#      TRIG.C
#
sed 's/^X//' << 'SHAR_EOF' > ATOF.C &&
X/* ATOF function for Sozobon C.                          */
X/* Copyright = David Brooks, 1989 All Rights Reserved
         */
X
X/* Hack warning.  Hack warning.  Hack warning...
     */
X/* Somehow I feel scanf should call atof, not the other way around.
     */
X/* Guys???                                      */
X
Xfloat atof(s)
Xchar *s;
X{   float    res;
X
X    sscanf(s, "%g", &res);
X    return res;
X}
X
X_scandum()
```

```
X{
X    _scanf();          /* Force floating version to load */
X}
SHAR_EOF
chmod 0600 ATOF.C || echo "restore of ATOF.C fails"
sed 's/^X//' << 'SHAR_EOF' > EXP.C &&
X/* EXP and hyperbolic trig functions for Sozobon C.
         */
X/* Copyright = David Brooks, 1989 All Rights Reserved
         */
X
X#include <errno.h>
X
Xtypedef union {
X    unsigned long ul;
X    float f;
X    char sc[4];            /* Assuming char signed */
X    } fstruct;
X
X/* EXP:                                          */
X/*                                          */
X/* Uses the synthesis (2**n) * (2**[m/8]) ** b**g
     */
X/* where b = 2**(1/8) and 8*n+m+g = arg/ln(b).  b**g has a cubic
     */
X/* approximation: source and accuracy unknown.
         */
X/*                                          */
X/* Beware a Sozobon C bug: can't compare two negative floating numbers.
     */
X
Xfloat exp(a) register float a;
X{   fstruct       res;
X    register int  aint;
X
X    static fstruct huge = 0xFFFFFF7FL;
X    static float powtab[] = {1.0, 1.09050773, 1.1892071, 1.2968396,
X              1.41421356, 1.5422108, 1.68179283, 1.8340081};
X
X    if (a > 43.5)
X    {    errno = EDOM;
X         return huge.f;
X    }
X
X    if ((a + 43.5) < 0.0)
X         return 0.0;                 /* Underflow */
X
X    if ((aint = (int)(a *= 11.5415603)) < 0)  /* 8/ln(2) */
X         --aint;                /* Correct mathematically */
X    a = (float)aint - a;             /* -(frac part) */
X    res.f = 1.0 + a * (-0.0866439378 + a * (0.003750577 +
X                  a * -0.11321783e-3));
X    res.sc[3] += aint >> 3;          /* Mult by 2**n
```

```
     */
X    return res.f * powtab[aint&7];       /* Mult by 2**[m/8]
     */
X}
X
X/* HYPERBOLIC FUNCTIONS: virtually free
     */
X
Xfloat sinh(a) float a;
X{   float    exp();
X    register float     ea = exp(a);
X
X    return 0.5 * (ea - (1.0 / ea));
X}
X
Xfloat cosh(a) float a;
X{   float    exp();
X    register float     ea = exp(a);
X
X    return 0.5 * (ea + (1.0 / ea));
X}
X
Xfloat tanh(a) float a;
X{   float    exp();
X    register float     e2a;
X
X    e2a = exp(a);
X    e2a = e2a * e2a;            /* exp-squared-a */
X    return (e2a - 1.0) / (e2a + 1.0);
X}
X
SHAR_EOF
chmod 0600 EXP.C || echo "restore of EXP.C fails"
sed 's/^X//' << 'SHAR_EOF' > FABS.C &&
X/* FABS, LDEXP, and FREXP functions for Sozobon C.
     */
X/* Copyright = David Brooks, 1989 All Rights Reserved
         */
X
X#include <errno.h>
X
Xtypedef union {
X    unsigned long ul;
X    float f;
X    char sc[4];            /* Assuming char signed */
X    } fstruct;
X
X/* FABS: directly manipulate the sign bit, punning the type.  Actually
     */
X/* the argument and return are both float.
     */
X
Xunsigned long fabs(a)
```

```
Xunsigned long a;
X{   return a & 0xFFFFFF7FL;}
X
X/* LDEXP: again direct manipulation.                    */
X
Xfloat ldexp(a, n)
Xfstruct a;
Xint n;
X{   int  exp = (a.sc[3] & 0x7F) + n;
X
X    if (exp < 0)
X         return 0L;              /* Underflow */
X
X    if (exp > 0x7F)
X    {    errno = EDOM;          /* Overflow */
X         return a.ul | 0xFFFFFF7FL;  /* Preserve sign */
X    }
X
X    a.sc[3] = (a.sc[3] & 0x80) | exp;
X    return a.f;
X}
X
X/* FREXP: split into convenient mantissa and exponent
          */
X
Xfloat frexp(a, ip)
Xfstruct a;
Xint *ip;
X{   if (a.ul == 0L)
X         *ip = 0;
X    else
X    {    *ip = (a.sc[3] & 0x7f) - 0x40;
X         a.sc[3] = (a.sc[3] & 0x80) | 0x40;    /* preserve sign,
X                                    force exponent */
X    }
X
X    return a.f;
X}
SHAR_EOF
chmod 0600 FABS.C || echo "restore of FABS.C fails"
sed 's/^X//' << 'SHAR_EOF' > FMOD.C &&
X/* CEIL, FLOOR, MODF, and FMOD functions for Sozobon C.
              */
X/* Copyright = David Brooks, 1989 All Rights Reserved
          */
X
Xtypedef union {
X    unsigned long ul;
X    float f;
X    char sc[4];             /* Assuming char signed */
X    } fstruct;
X
X/* This macro returns true if the number is so large that is has no
```

```
     */
X/* representable fraction.                        */
X
X#define ISLGINT(a)    (((a).sc[3] & 0x7F) > (0x40 + 24))
X
X/* CEIL: smallest integral value not less than a.
     */
X
Xfloat ceil(a)
Xfstruct a;
X{   register float    aint;
X
X    if (ISLGINT(a))
X         return a.f;
X
X    if ((aint = (float)((long)a.f)) != a.f && a.sc[3] >= 0)
X         ++aint;
X
X    return aint;
X}
X
X/* FLOOR: largest integral value not greater than a.
          */
X
Xfloat floor(a)
Xfstruct a;
X{   register float    aint;
X
X    if (ISLGINT(a))
X         return a.f;
X
X    if ((aint = (float)((long)a.f)) != a.f && a.sc[3] < 0)
X         --aint;
X
X    return aint;
X}
X
X/* Just for once, we luck out: negative numbers behave as needed... */
X
Xfloat modf(a, ip)
Xfstruct a;
Xfloat *ip;
X{   long ipart;
X
X    *ip = (ISLGINT(a))?a.f:(float)(long)a.f;
X
X    return a.f - *ip;
X}
X
Xfloat fmod(n, d)
Xfloat n, d;
X{   fstruct quot;
X    long iquot;
```

```
X
X    quot.f = n / d;
X    if (ISLGINT(quot))
X         return 0.0;
X
X    iquot = (long)quot.f;
X    return n - (float)iquot * d;
X}
SHAR_EOF
chmod 0600 FMOD.C || echo "restore of FMOD.C fails"
sed 's/^X//' << 'SHAR_EOF' > FPCMP.S &&
X* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X*
X* Permission is granted to anyone to use this software for any purpose
X* on any computer system, and to redistribute it freely, with the
X* following restrictions:
X* 1) No charge may be made other than reasonable charges for reproduction.
X* 2) Modified versions must be clearly marked as such.
X* 3) The authors are not responsible for any harmful consequences
X*    of using this software, even if they result from defects in it.
X*
X* MODIFICATIONS:
X*
X* D.W.Brooks 3/5/89   Fixed fpcmp (when both -ve) and fpneg (when zero)
X*
X
X    .globl   fpcmp
X    .globl   _fpcmp
Xfpcmp:
X_fpcmp:
X    move.l   4(sp),d0
X    move.l   8(sp),d1
X* Start DWB
X    move.b   d0,d2         * See if both signs are negative
X    and.b    d1,d2
X    bpl  L0
X    move.l   d0,d2         * They are: switch arguments so the rest works
X    move.l   d1,d0
X    move.l   d2,d1
XL0:
X* End DWB
X    cmp.b    d1,d0
X    bne  L1
X    cmp.l    d1,d0
XL1:
X    rts
X
X    .globl   fpneg
X    .globl   _fpneg
Xfpneg:
X_fpneg:
X    move.l   4(sp),d0
X
```

```
X* DWB removed: tst.b  d0
X* This would have not negated very small numbers.
X
X    beq  L2
X    eor.b    #$80,d0
XL2:
X    rts
SHAR_EOF
chmod 0600 FPCMP.S || echo "restore of FPCMP.S fails"
sed 's/^X//' << 'SHAR_EOF' > ITRIG.C &&
X/* ASIN, ACOS, ATAN, and ATAN2 functions for Sozobon C.
              */
X/* Copyright = David Brooks, 1989 All Rights Reserved
         */
X
X#include <errno.h>
X
X#define PI   3.14159265
X#define PIBY2    1.57079633
X
Xtypedef union {
X    unsigned long ul;
X    float f;
X    char sc[4];            /* Assuming char signed */
X    } fstruct;
X
X/* ARCTAN:                                      */
X/*                                        */
X/* Single precision is 24 bits, or better than 7 decimal digits.  The
     */
X/* method is from Hart et al: "Computer Approximations".  It uses a
     */
X/* divide-and-conquer algorithm.  The argument range is divided into
     */
X/* five using the partition points tan({1,3,5,7}*pi/16), a polynomial
     */
X/* valid over +/- pi/16 is calculated, and other magic is used to
     */
X/* reposition the result.  Precision is >8 places.
     */
X
Xfloat atan(a) fstruct a;
X{   fstruct       absval;
X    register float    tx0, tsq, temp;
X    register int  part;
X    register char sign;
X
X    static float  adj[] = {0.0, 0.414213562, 1.0, 2.414213562};
X    static float  atof[] = {0.0, 0.39269908, 0.78539816,
X                        1.178097245, 1.57079633};
X
X    sign = a.sc[3] & 0x80;
X    a.sc[3] &= 0x7f;            /* get fabs(a) */
```

```
X    tx0 = a.f;
X
X/* Figure out the partition */
X
X    part = tx0<0.66817864?(tx0<0.19891237?0:1):(tx0<1.49660576?2:
X              tx0<5.0273395?3:4);
X    if (part == 4)
X         tx0 = -1.0 / tx0;
X    else if (part != 0)
X    {    temp = adj[part];
X         tx0 = (tx0 - temp) / (tx0 * temp + 1.0);
X    }
X
X/* Here's the calculation */
X
X    tsq = tx0 * tx0 + 1.67784279;
X    a.f = (0.93833093 / tsq + 0.44075154) * tx0 + atof[part];
X
X    a.sc[3] ^= sign;            /* Negate if negative */
X    return a.f;
X}
X
X/* ARCSIN and ARCCOS use the standard identities.  There's some less
     */
X/* optimal code here because Sozobon C can't properly compare two
     */
X/* negative floating numbers.                          */
X
Xfloat asin(a) float a;
X{   float    atan(), sqrt();
X
X    if (a > 1.0 || (a+1.0) < 0.0)
X    {    errno = EDOM;
X         return 0.0;
X    }
X
X    return atan(a / sqrt(1.0 - a*a));
X}
X
Xfloat acos(a) float a;
X{   float    atan(), sqrt();
X    float    temp;
X
X    if (a > 1.0 || (a+1.0) < 0.0)
X    {    errno = EDOM;
X         return 0.0;
X    }
X
X    temp = atan(sqrt(1.0 - a*a) / a);
X    if (a >= 0.0)
X         return temp;
X    else
X         return PI + temp;
```

```
X}
X
X/* ATAN2:                                          */
X/*                                       */
X/* Computes atan(quotient), and returns that for positive cos, else
     */
X/* extends the range depending on the sin.
     */
X
Xfloat atan2(s, c) fstruct s, c;
X{   register float     r;
X    float         atan();
X
X    if (c.sc[3] == 0L)          /* Infinite argument */
X         return (s.sc[3]<0)?-PIBY2:PIBY2;
X
X    r = atan(s.f / c.f);
X
X    if (c.sc[3] >= 0)
X         return r;          /* Range -pi/2..pi/2 */
X    if (s.sc[3] >= 0)
X         return PI + r;
X    return -PI + r;
X}
SHAR_EOF
chmod 0600 ITRIG.C || echo "restore of ITRIG.C fails"
sed 's/^X//' << 'SHAR_EOF' > LOG.C &&
X/* LOG functions for Sozobon C.                         */
X/* Copyright = David Brooks, 1989 All Rights Reserved
          */
X
X#include <errno.h>
X
Xtypedef union {
X    unsigned long ul;
X    float f;
X    char sc[4];            /* Assuming char signed */
X    } fstruct;
X
X/* LOG:                                          */
X/*                                       */
X/* This yanks out the exponent, performs a polynomial valid over +/-
     */
X/* sqrt(2), and adds the scaled exponent back in.  Domain error on
     */
X/* arguments <= 0.                                  */
X
Xfloat log(a) fstruct a;
X{   register float     t1, t2, t3;
X    static fstruct nhuge = 0xFFFFFFFFL;  /* Largest negative
     */
X
X    if (a.f <= 0.0)
```

```
X    {    errno = EDOM;          /* Impossible really   */
X         return nhuge.f;
X    }
X
X    if (a.f < 1.41421356 && a.f > 0.70710678)
X    {    t3 = 0.0; /* arg between sqrt(2) and sqrt(2)/2 */
X         t2 = (t1 = a.f - 1.0) + 2.0;
X    }
X    else
X    {    t3 = (float)(((int)a.sc[3] << 1) - 0x81) * 0.34657359;
X              /* ...(log2(a) * 2 + 1) * ln(2)/2 */
X         a.sc[3] = 0x40;             /* Scale back to range */
X         t2 = a.f + 0.70710678;
X         t1 = a.f - 0.70710678;
X    }
X
X    t1 /= t2;
X    t2 = t1 * t1;
X
X    return t3 + t1 * (2.0000008 + t2 * (0.66644078 + t2 * 0.41517739));
X}
X
X/* LOG10 is easy. */
X
Xfloat log10(a) float a;
X{   float    log();
X    return 0.43429448 * log(a); /* ln(a) * log10(e) */
X}
SHAR_EOF
chmod 0600 LOG.C || echo "restore of LOG.C fails"
sed 's/^X//' << 'SHAR_EOF' > MATH.H &&
X/*
X *  MATH.H        Math function declarations
X */
X
X#ifndef MATH_H
X#define  MATH_H
X
Xextern  float    sin();
Xextern  float    cos();
Xextern  float    tan();
Xextern  float    asin();
Xextern  float    acos();
Xextern  float    atan();
Xextern  float    atan2();
Xextern  float    sinh();
Xextern  float    cosh();
Xextern  float    tanh();
Xextern  float    exp();
Xextern  float    log();
Xextern  float    log10();
Xextern  float    pow();
Xextern  float    sqrt();
```

```
Xextern  float    ceil();
Xextern  float    floor();
Xextern  float    fabs();
Xextern  float    ldexp();
Xextern  float    frexp();
Xextern  float    modf();
Xextern  float    fmod();
Xextern  float    atof();
X
X#endif MATH_H
SHAR_EOF
chmod 0600 MATH.H || echo "restore of MATH.H fails"
sed 's/^X//' << 'SHAR_EOF' > POW.C &&
X/* POW function for Sozobon C.                          */
X/* Copyright = David Brooks, 1989 All Rights Reserved
         */
X/*                                        */
X/* This uses exp and log in the obvious way.  A domain error occurs if
     */
X/* a=0 and p<=0, or a<0 and p is not integral.
         */
X
X#include <errno.h>
X
Xtypedef union {
X    unsigned long ul;
X    float f;
X    char sc[4];            /* Assuming char signed */
X    } fstruct;
X
Xfloat pow(a, p)
Xfstruct a;
Xregister float p;
X{   fstruct       r;
X    float         exp(), log();
X    register long pint;
X    register int  sign = 0;
X
X    if (a.ul == 0L && p <= 0.0)
X    {    errno = EDOM;
X         return 0.0;
X    }
X
X    if (a.sc[3] < 0)       /* ...if a.f < 0 */
X    {    pint = (long)p;
X         if ((float)pint != p)  /* ... if nonintegral */
X              errno = EDOM;
X         sign = pint & 1;   /* ... carry on anyway */
X         a.sc[3] &= 0x7f;
X    }
X
X    r.f = exp(log(a.f) * p);
X    if (sign)
```

```
X        r.sc[3] |= 0x80;
X    return r.f;
X}
SHAR_EOF
chmod 0600 POW.C || echo "restore of POW.C fails"
sed 's/^X//' << 'SHAR_EOF' > README &&
X        FLOATING POINT LIBRARY FOR SOZOBON C
X
X        Rev 1.0 -- prepared March 6, 1989
X
XThe files contained in this archive are an initial release of some
Xfloating point functions for Sozobon C.  We provide:
X
X    sqrt     exp      fabs
X    sin      log      ldexp
X    cos      log10         frexp
X    tan      pow      atof
X    atan     ceil
X    atan2         floor         math.h
X    asin     modf
X    acos     fmod
X
XThese are available in source.  There is also a new LIBM.A that has the
Xnew routines included (and the other routines re-arranged to remove the
Xneed to scan the library twice).
X
XThese routines may be used without restriction, but I retain the copyright
Xon them, and the Only True Upgrades will be those I make.  If you have any
Ximprovements, please send them to me.
X
XDavid W. Brooks
X36 Elm Street
XMedfield, MA 02052
X
XBROOKS@CSSS-A.PRIME.COM
X{uunet,mit-eddie}!csss-a.prime.com!brooks
SHAR_EOF
chmod 0600 README || echo "restore of README fails"
sed 's/^X//' << 'SHAR_EOF' > SQRT.C &&
X/* SQRT function for Sozobon C.                          */
X/* Copyright = David Brooks, 1989 All Rights Reserved
         */
X/*                                          */
X/* Single precision is 24 bits, or better than 7 decimal digits.  The
     */
X/* method is from Hart et al: "Computer Approximations".  We reduce
     */
X/* the range to [0.25..1.0], then apply the polynomial:
              */
X/* 0.25927688 + 1.0520212n - 0.31632214n^2, giving 2.3 digits
         */
X/* approximation.  Then we do two Newton's iterations (giving ~9 digits
     */
```

```
X/* precision.                                          */
X/*                                      */
X/* A negative argument gives a return value of 0.0 and sets EDOM.
     */
X
X#include <errno.h>
X
Xtypedef union {
X    unsigned long ul;
X    float f;
X    char sc[4];            /* Assuming char signed */
X    } fstruct;
X
X/* Sadly we can't use register for objects of type fstruct. */
X
Xfloat sqrt(n)
Xfstruct n;
X{   register int exp;      /* Separate unbiased exponent.
     */
X    fstruct s;
X
X    if ((n.ul & 0xFFFFFF00L) == 0L)  /* Check for unnormalized 0.
     */
X        return 0.0;
X
X    if ((exp = n.sc[3]) < 0)    /* Test argument's sign.
     */
X    {   errno = EDOM;
X        return 0.0;
X    }
X
X    exp -= 0x40;           /* Get unbiased exponent.   */
X    if (exp & 1)
X    {   n.sc[3] = 0x3F;        /* Convert to 0.25..0.5.
     */
X        ++exp;
X    }
X    else
X        n.sc[3] = 0x40;        /* Or convert to 0.5..1.
     */
X
X    s.f = 0.25927688 + n.f * (1.0520212 + n.f * -0.31632214);
X
X    s.f = s.f + n.f / s.f;      /* One Newton.
     */
X    s.sc[3] -= 1;          /* (here's the divide by 2).
     */
X    s.f = s.f + n.f / s.f;      /* The other Newton.
     */
X    s.sc[3] += ((unsigned int)exp >> 1) - 1;
X                      /* Divide by 2 and insert
X                         half the original exponent.   */
X    return s.f;
```

```
X}
SHAR_EOF
chmod 0600 SQRT.C || echo "restore of SQRT.C fails"
sed 's/^X//' << 'SHAR_EOF' > TRIG.C &&
X/* SIN, COS, and TAN functions for Sozobon C.
     */
X/* Copyright = David Brooks, 1989 All Rights Reserved
          */
X
X#include <errno.h>
X
X#define PIBY2     1.57079633
X#define TWOBYPI   0.636619772
X
Xtypedef union {
X    unsigned long ul;
X    float f;
X    char sc[4];            /* Assuming char signed */
X    } fstruct;
X
X/* SIN:
X/*                                        */
X/* Single precision is 24 bits, or better than 7 decimal digits.  The
     */
X/* method is from Hart et al: "Computer Approximations".  It reduces
     */
X/* the range to 0..pi/2, scaled to 0..1, and applies:
          */
X/* 1.57079629t - 0.64596336t^3 + 0.0796884805t^5 - 0.0046722279t^7
     */
X/*       + 0.00015082056t^9 (precision of 8.5 digits) and
     */
X/* finally fixes the sign.                          */
X
Xfloat sin(a) fstruct a;
X{   register float         t, ipart, tsq;
X    fstruct           x;
X    register unsigned long tint;
X    register char     sign;
X
X    sign = a.sc[3] & 0x80;
X    a.sc[3] ^= sign;       /* sin(negative x) = -sin(-x) */
X
X/* If the argument is huge, the result is rather arbitrary -- and,
     */
X/* besides, the following code will break.
     */
X
X    if (a.sc[3] >= (64+24))
X    {    errno = EDOM;
X         return 0.0;
X    }
X
```

```
X    t = a.f * TWOBYPI;      /* Express in quarter-turns */
X
X/* Get integer part and fraction.  Optimise the common case, in the
X   first quadrant */
X
X    if (t <= 1.0)
X         tint = 0L;
X    else
X    {    tint = t;          /* tint = whole quarterturns */
X         ipart = (float)tint;
X         t = (tint & 1)?ipart - t + 1.0:t - ipart;   /* Get fraction */
X    }
X    tsq = t * t;
X    x.f = t * (1.57079629 + tsq * (-0.64596336 + tsq * (0.0796884805
X         + tsq * (-0.46722279e-2 + tsq * 0.15082056e-3))));
X
X    if ((int)tint & 2)          /* Quadrants 3 and 4    */
X         x.sc[3] |= 0x80;       /* Set negative         */
X    if (sign != 0)
X         x.sc[3] ^= 0x80;       /* Negate     */
X
X    return x.f;
X}
X
X/* COS:                                          */
X/*                                        */
X/* Although approximations are available, this will do fine.
          */
X
Xfloat cos(a) float a;
X{   float    sin();
X    return sin(PIBY2 - a);
X}
X
X/* TAN:                                          */
X/* (v-e-r-y s-l-o-w-l-y)                              */
X
Xfloat tan(a) float a;
X{   float    sin();
X    return sin(a) / sin(PIBY2 - a);
X}
SHAR_EOF
chmod 0600 TRIG.C || echo "restore of TRIG.C fails"
exit 0
```

```
From 4988493827165587114
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,fb90ba50afa0f287,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-29 09:55:24 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!csd4.milw.wisc.edu!bionet!agate!
saturn!ssyx.ucsc.edu!koreth
```

From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i034: prtset -- Set printer parameters
Keywords: shar, 68000
Message-ID: <6800@saturn.ucsc.edu>
Date: 29 Mar 89 17:55:24 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 464
Approved: koreth@ssyx.ucsc.edu

Submitted-by: ntomczak@ualtavm.bitnet (Michal Jaegermann)
Posting-number: Volume 2, Issue 34
Archive-name: prtset

[Ha! After weeks of trying, we've beaten you, BITNET! I finally got this
 intact and unmangled! Hahahhahahahahahahahaha.... Oh, sorry. -sg]

This is a small utility, meant mostly for shell scripts, but also for
an interactive use, which main purpose is to send all kind of weird
initialization codes to a printer. For example, when I am printing
.dvi files produced by TeX I am using the following two Gulam aliases,
which include some of MY printer magic, initialize system and printer
properly and restore printer to a sane (more or less) state after dvi
driver is finished.

```
dvi  prtset -s -h1b\8 ; dvc
dvc  pushd d:\texbin ; gem dvi.prg ; prtset -c -h1b401b\M1b\90d ; popd
```

I am using command 'dvi' to start my previewer, with printer off, or to
print on a single-sheet paper, and command 'dvc' to print on a
continous one. The other role of a prtset program is to set and check
proper configuration codes in situation when keeping around in memory
fat control.acc is not exactly what you would really want to do. When
printer is not on line 'prtset' just returns. It is written in
assembler, since I wanted it to be really small. I bet that you will
find also some other ways to use it.

   Enjoy,

   Michal Jaegermann
   Myrias Research Corporation
   Edmonton, Alberta, CANADA
   myrias!mj@alberta

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      PRTSET.DOC
#      PRTSET.S
#
sed 's/^X//' << 'SHAR_EOF' > PRTSET.DOC &&
X
```

X
X
XPRTSET   ST USER MANUAL   PRTSET
X
X
XNAME
X   prtset - a program to set, change and display system printer
X   information and to send printer control codes
X
XSYNOPSIS
X   prtset [-c][-o][-s][-n<infocode>][-h<printcode>]
X
XDESCRIPTION
X   Prtset is a small program for a control of printing by mody-
X   fying a system printer information and sending printer codes
X   to your printer. It is meant mostly for situations in which
X   you have better uses for memory then keeping around a bulky
X   accesory. It can be used interactively, but the best place
X   for it is probably in shell scripts and command aliases.
X   Contrary to what you may think some programs do care how the
X   printer information was set.
X
X   The program checks if an printing device is on line. If it
X   is not, then an attempt to send something to the printer will
X   be silently skipped, and prtset will return, possibly
X   performing other requested actions.
X
XOPTIONS
X   The following are valid command-line options:
X
X   -n<infocode> No space between flag and code allowed.
X   Set printer information accordingly to infocode.
X   An infocode is a two-digit hex number with a bit
X   pattern set with a help of the following table:
X
X   bit   reset   set
X   --------------------------------
X   0   dot-matrix   daisy-wheel
X   1   b/w   colour
X   2   1240 dots/l   960 dots/l
X   3   draft   final
X   4   parallel   serial
X   5   continuous   single-sheet
X   6-15   reserved
X
X   -o   Show an old configuration. An output is a string
X   of 0's and 1's corresponding to reset and set bits
X   in a system printer information.
X
X   -c   Switch to printing on countinuous paper
X
X   -s   Switch to printing on single-sheet paper
X

```
X     -h<printcode>     Send to the printer a string of characters
X             defined by printcode. Characters in printcode are
X         represented either as their two-digit hexadecimal
X         codes, in either case, or literally as themselves, if
X         preceded by an escape charater '\'.  Any non-escaped
X         character, which is not a hexadecimal digit, will ter-
X         minate printcode.  Remaining non-white space characters
X         will be skipped.
X
XEXAMPLE
X    Configure a printer to Epson in NLQ mode on RS-232 port printing
X    on a fan-fold paper and set right print margin to 65 characters
X    (your control sequence may vary, check your printer manual).
X
X             prtset -n1c -h1b\Q410d
X
X    Display old printer configuration and switch to printing on
X    a single sheet paper
X
X         prtset -o -s
X
XAUTHOR
X     prtset was written by Michal Jaegermann, September 1988.
X
XBUGS
X     A command line is processed sequentially from left to right.
X     Therefore the following will work:
X
X             prtset -o -n2a -o -n1c -o -n00 -o
X
X     displaying printer configuration four times.  The same applies
X     to all other options.  This may be construed as a bug, or as
X     a feature - depending on a state of your mind.
X
X     Due to a well-know brain damage in a desktop the following,
X     executed from a command window,
X
X             prtset -h\t\e\s\t0d0a
X
X     will print 'TEST', instead of 'test'.  Use a shell which will
X     leave your command line alone, or replace escaped lower case
X     characters with corresponding hexadecimal codes.
SHAR_EOF
chmod 0600 PRTSET.DOC || echo "restore of PRTSET.DOC fails"
sed 's/^X//' << 'SHAR_EOF' > PRTSET.S &&
XFrom @CORNELLC.ccs.cornell.edu:NTOMCZAK@UALTAVM Wed Mar 29 00:34:11 1989
XReceived: from CORNELLC.ccs.cornell.edu ([128.253.1.19]) by ssyx.ucsc.edu
(4.0/1.1)
X    id AA06846; Wed, 29 Mar 89 00:34:04 PST
XMessage-Id: <8903290834.AA06846@ssyx.ucsc.edu>
XReceived: from UALTAVM by CORNELLC.ccs.cornell.edu (IBM VM SMTP R1.2.1MX) with
BSMTP id 0401; Wed, 29 Mar 89 03:33:47 EST
XReceived: by UALTAVM (Mailer X1.25) id 6719; Wed, 29 Mar 89 01:32:37 MST
```

```
XDate:         Wed, 29 Mar 89 01:30:02 MST
XFrom: Michal Jaegermann <NTOMCZAK%UALTAVM.BITNET@CORNELLC.CIT.CORNELL.EDU>
XSubject:      Re:  prtset
XTo: Steven Grimm <koreth@ssyx>
XIn-Reply-To:  Your message of Tue, 28 Mar 89 21:58:33 PST
XStatus: R
X
X     Line noise was actually quite bad. :-).  Ok, here is the rest
X-----------------------------------------------------------------------
XSteve,
X
X   I've already seen posted other parts of this mailing.  This file
Xhas to be the most persistently refusing to transfer which I ever
Xseen. Hope that this clear ASCII text will get to you in more or less
Xone piece.
X
X   Regards,
X   Michal
X
X------- cut here ----------------
X
X*****************************************************************
X*                                   *
X*    prtset.s   version 1.1                      *
X*    Michal Jaegermann, September 1988                *
X*                                   *
X*    this software is released to a public domain -
     *
X*    enjoy every way you like                    *
X*                                                          *
X*****************************************************************
X
X* character codes
Xtab      equ   9
Xlf       equ   10
Xcr       equ   13
X
X* gemdos & (x)bios equates
Xgemdos   equ   1
Xbios     equ   13
Xxbios    equ   14
X
Xconin    equ   1
Xconout    equ   2
Xauxout    equ   4
Xprnout   equ   5
Xpoutst    equ   $11
Xaoutst   equ   $13
Xcconws   equ   9
Xpterm    equ   $4c
Xsetprt    equ   33
X
X    .text
```

```
Xinit:
X*   move.w   #-1,d0        ; find current printer configuration
X    bsr  oldcnf
X    move.w   d0,prcnfg
X    move.w   #prnout,prdev
X    btst #4,d0
X    beq.s    getarg
X    move.w   #auxout,prdev
Xgetarg:
X    move.l   4(sp),a6
X    adda.w   #$80,a6       ; a6 points to argument string start
X    moveq.l  #0,d0
X    move.b   (a6)+,d0 ; Get string length and
X    clr.b    (a6,d0.w) ; ensure '\0' termination
X    sf   somearg
Xnxtarg:
Xskipwh:
X    bsr  getch
Xtestch:
X    tst.b    d0
X    beq  done      ; '\0' char is end of string
X    cmp.b    #' ',d0
X    beq  skipwh
X    cmp.b    #tab,d0
X    beq  skipwh
X    cmp.b    #'-',d0
X    beq  skipwh
X*
X    st   somearg
Xopto:
X    cmp.b    #'O',d0
X    bne.s    optn
X    bsr  shwcnf
X    bra.s    nxtarg
Xoptn:
X    cmp.b    #'N',d0
X    bne.s    optc
X    bsr  newcnf
X    bra.s   nxtarg
Xoptc:
X    cmp.b    #'C',d0       ; using continuous paper
X    bne.s    opts
X    move.w   prcnfg,d0
X    bclr #5,d0
X    bsr  prconf
X    bra.s    nxtarg
Xopts:
X    cmp.b    #'S',d0       ; using single-sheet
X    bne.s    opthx
X    move.w   prcnfg,d0
X    bset #5,d0
X    bsr  prconf
X    bra.s    nxtarg
```

```
X
Xopthx:
X    cmp.b    #'H',d0       ; hex escape
X    bne.s    usage
X    bsr  sendc         ; return code quietly ignored
X    bra.s    testch
X*
Xdone:
X    tst.b    somearg
X    beq.s    usage
X    move.w   #0,-(sp)  ; clear status
X    bra.s    term
Xusage:
X    pea.l    usemsg        ; If error in arg string show usage message
X    move.w   #cconws,-(sp)
X    trap #gemdos
X    addq.l   #6,sp
X    move.w  #conin,-(sp)    ; wait for keypress to mollify desktop
X    trap #gemdos
X*   addq.l   #2,sp         ; conin code doubles as an error status
X*   move.w   #1,-(sp)  ; error status
Xterm:
X    move.w   #pterm,-(sp)
X    trap #gemdos       ; and terminate
X
Xgetch:
X*                ; return in d0 next character
X*                ; converting lower case to upper
X*                ; condition code set EQ if char is NULL
X    move.b   (a6)+,d0
X    cmp.b    #'a',d0
X    blt.s    gcrtn
X    cmp.b    #'z',d0
X    bgt.s    gcrtn
X    eori.b   #$20,d0       ; convert to uppercase
Xgcrtn:
X    tst.b    d0       ; set condition codes
X    rts
X
X*                ; set new printer configuration
Xprconf:
X    move.w   d0,-(sp)
X    andi.w  #$3f,d0        ; only these bits meanigful
X    move.w  prcnfg,d1  ; do not change reserved bits
X    andi.w  #%0111111111000000,d1
X    or.w    d1,d0
X    move.w   #setprt,-(sp)
X    trap #xbios
X    addq.l   #4,sp
Xoldcnf:
X    moveq    #-1,d0        ; check printer configuration
X    move.w  d0,-(sp)
X    move.w   #setprt,-(sp)
```

```
X    trap #xbios
X    addq.l    #4,sp
X    move.w    d0,prcnfg ; store result in prcnfg
X    move.w    #prnout,prdev ; assume printer on parallel port
X    btst #4,d0          ; it this true?
X    beq.s     pcfret
X    move.w    #auxout,prdev    : no? - change output to aux
Xpcfret:
X    rts
X
Xcvthx:
X*       converts ascii character in d0 into hex digit left in d0
X*       sets HI condition code iff character is not convertible
X
X    subi.b    #'0',d0
X    cmpi.b    #9,d0
X    ble.s     cvret
X    subq.b    #7,d0          ; diff between '9' and 'A'
X    cmp.b     #$0f,d0
Xcvret
X    rts
X
Xsendc:
X*     sends to a printer hex codes which follow option -h
X*     trailing single digit codes ok
X*     characters escaped with \ represent their own ascii codes
X*     string is terminated by any not escaped non-hex character
X*     and its remaining part skipped until blank is reached
X*     -- sendc returns the folowing info in register d1
X*        -1 - codes send succesfully
X*         0 - cannot send to output device
X*         1 - output device is off-line
X*
X*     Warning: lower case escaped characters will not work from
X*     a desktop, due to a brain damaged upper case conversion by GEM.
X*
X*             ; check if otput device is available
X    moveq     #1,d1
X    move.w    #poutst,-(sp)
X    cmpi.w  #prnout,prdev
X    beq.s     chkdv
X    addq.w  #2,(sp)  ; ask for status of rs232
Xchkdv:
X    trap #gemdos
X    addq.l    #2,sp
X    tst.w     d0
X    beq.s   skips
Xsndnxt:
X    bsr  getch
X    beq.s     sret
X    moveq     #0,d1
X    cmp.b   #'\',d0
X    bne.s     cnvt1
```

```
X    move.b  (a6)+,d0
X    beq.s    sret
X    move.b   d0,d1
X    bra.s    print0
Xcnvt1:
X    bsr.s    cvthx
X    bhi.s    skips
X    move.b   d0,d1
X    bsr  getch
X    bne.s    nxtcv
X    bsr  printc
X    moveq    #0,d0
X    bra.s    sret
Xnxtcv:
X    bsr.s    cvthx
X    bhi.s    prnt1
X    lsl.b    #4,d1
X    add.b    d0,d1
Xprint0:
X    bsr  printc
X    beq.s    skips
X    bra.s    sndnxt
Xprnt1:
X    bsr  printc
Xskips:
X    bsr  getch
X    beq.s   sret
X    cmpi.b  #' ',d0
X    bne.s    skips
X    cmpi.b   #tab,d0
X    bne.s    skips
Xsret:
X    rts
X
Xprintc:
X    move.w  d1,-(sp)
X    move.w  prdev,-(sp)
X    trap #gemdos
X    addq.l   #4,sp
X    move.w  d0,d1      ; move result flag to d1
X    rts
X
Xshwcnf:
X    move.w   prcnfg,d2
X    lsl.w   #8,d2
X    lsl.w   #2,d2
X    moveq    #5,d3
X    move.w  #'0',d4
Xnxtbit:
X    move.w   d4,-(sp)
X    lsl.w    #1,d2
X    bcc.s    show
X    addq.w   #1,(sp)
```

```
Xshow:
X    move.w   #conout,-(sp)
X    trap #gemdos
X    addq.l   #4,sp
X    dbra d3,nxtbit
X    pea.l    retmsg
X    move.w   #cconws,-(sp)
X    trap #gemdos
X    addq.l   #6,sp
X    rts
X
Xnewcnf:
X    bsr  getch
X    beq.s    unget
X    bsr  cvthx
X    bhi.s    nret
X    move.b   d0,d1
X    bsr  getch
X    beq.s    unget
X    bsr  cvthx
X    bhi.s    nret
X    lsl.b    #4,d1
X    add.b    d0,d1
X
X    move.w  d1,d0
X    bclr #15,d0
X    bsr  prconf
Xnret:
X    rts
Xunget:
X    subq.l #1,a6  ; we will test terminator once again
X    rts
X
X    .data
X
Xusemsg: dc.b  'usage: prtset [-c|-s] [-o|-n<hex>] [-h<hex>]',cr,lf
X    dc.b  tab,'-c|-s',tab,'continuous | single-sheet',cr,lf
X    dc.b  tab,'-o|-n',tab,'old | new configuration',cr,lf
X    dc.b  tab,'-h<hex>',tab,'codes to send',cr,lf
X    dc.b  tab,tab,'where codes are hex or \escaped literals'
Xretmsg: dc.b  cr,lf,0
Xsomearg:
X    dc.b  0
X
X    .even
X    .bss
Xrpcnfg: ds.w  1
Xprdev:  ds.w  1
X    end
X
SHAR_EOF
chmod 0600 PRTSET.S || echo "restore of PRTSET.S fails"
exit 0
```

From 5090977599727506513
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,6306438efeab1bd9,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-03-29 15:54:59 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!csd4.milw.wisc.edu!bionet!agate!
saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i035: hexify -- Hexidecimal encode/decode for mailing binaries
Keywords: shar, C
Message-ID: <6801@saturn.ucsc.edu>
Date: 29 Mar 89 23:54:59 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 145
Approved: koreth@ssyx.ucsc.edu

Submitted-by: ntomczak@ualtavm.bitnet (Michal Jaegermann)
Posting-number: Volume 2, Issue 35
Archive-name: hexify

[Uuencode is much more efficient, but you can use this to send binaries
 across troublesome networks, such as BITNET (sometimes). -sg]

Here are two C programs which I used to perform a conversion to hex
and back. Binary restorer is smart enough to find a "BEGIN" line and
create file name. Both programs will compile with Sozobon C and dLibs
library (and many other compilers, I bet). [SunOS 4.0's, for one... -sg]

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      dehex.c
#      hexify.c
#
sed 's/^X//' << 'SHAR_EOF' > dehex.c &&
X/*
X *  Hex-to-binary utility -- conquer BITNET mailers!!
X *
X *  Restore binary file from a text file with hexadecimal digits
X *
X *  Michal Jaegermann, 19 March 1989
X */
X
X#include <stdio.h>
X#include <string.h>
X#define  BSIZE 128
X
Xmain(argc, argv)
Xint argc;
```

```
Xchar **argv;
X{
X    int ch, count = 0;
X    FILE *fin, *fout;
X    char inbuf[BSIZE+2];
X    char outname[14];
X
X    if (argc != 2) {
X         fprintf(stderr, "usage: %s <infile>\n", argv[0]);
X         exit(1);
X    }
X
X    if (NULL == (fin = fopen(argv[1],"r"))){
X         fprintf(stderr, "cannot find input file %s\n", argv[1]);
X         exit(1);
X    }
X    /* search for BEGIN line */
X    for(;;) {
X         if (NULL == fgets(inbuf, BSIZE, fin)){
X              fprintf(stderr, "No BEGIN line\n");
X              fclose(fin);
X              exit(1);
X         }
X         sscanf(inbuf, " %12s", outname);
X         if (0 == strcmp("BEGIN", outname)) break;
X         if ('\n' != inbuf[strlen(inbuf) - 1]) {
X              while ('\n' != (ch = getc(fin))) {
X                   ; /* eat a remainder of a line */
X              }
X         }
X    }
X
X    /* get an output file name */
X    sscanf(inbuf, " %*5s %12s", outname);
X
X    if (NULL == (fout = fopen(outname,"wb"))){
X         fprintf(stderr, "cannot open output file %s", outname);
X         fclose(fin);
X         exit(1);
X    }
X
X    /* do the actual job */
X    while (1 == fscanf(fin, " %02x", &ch)) {
X         putc((char) ch, fout);
X    }
X    fclose(fout);
X
X    /* just a check for completeness */
X    fscanf(fin, " %3s", inbuf);
X    if (0 != strcmp("ND", inbuf)) { /* a little bit of cheating */
X         fprintf(stderr, "%s -- END line not found in %s\n",
X                   inbuf, argv[1]);
X    }
```

```
X    fclose(fin);
X    exit(0);
X}
X
SHAR_EOF
chmod 0600 dehex.c || echo "restore of dehex.c fails"
sed 's/^X//' << 'SHAR_EOF' > hexify.c &&
X/*
X *  Binary-to-hex utility -- conquer BITNET mailers!!
X *
X *  Rewrite binary file as a text file containing hexadecimal
X *  digits
X *
X *  Michal Jaegermann, 19 March 1989
X */
X
X#include <stdio.h>
X/* LWIDTH bytes --- will put twice as many characters on output line */
X#define LWIDTH 32
X
Xmain(argc, argv)
Xint argc;
Xchar **argv;
X{
X    int ch, count = 0;
X    FILE *fin, *fout;
X
X    if (argc != 3) {
X         fprintf(stderr, "usage: %s <infile> <outfile>\n", argv[0]);
X         exit(1);
X    }
X
X    if (NULL == (fin = fopen(argv[1],"rb"))){
X         fprintf(stderr, "cannot find input file %s", argv[1]);
X    }
X    if (NULL == (fout = fopen(argv[2],"w"))){
X         fprintf(stderr, "cannot open output file %s", argv[2]);
X    }
X
X    fprintf(fout, "BEGIN %s\n", argv[1]);
X    while (EOF != (ch = getc(fin))) {
X         fprintf(fout, "%02x", (unsigned) ch);
X         if (LWIDTH == (++count)) {
X              count = 0;
X              fprintf(fout,"\n");
X         }
X    }
X    fprintf(fout, "\nEND\n");
X    /*
X     * we do not care if we have an extra new line - restore
X     * utility will gobble it in any case
X     */
X    exit(0);
```

```
X}
X
SHAR_EOF
chmod 0600 hexify.c || echo "restore of hexify.c fails"
exit 0


From -5809523751385917454
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,11f0ab29457e722d,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-04-13 19:12:03 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i036: nutshell -- Use a shell from within a GEM prog
Keywords: shar, C
Message-ID: <6878@saturn.ucsc.edu>
Date: 14 Apr 89 02:12:03 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 717
Approved: koreth@ssyx.ucsc.edu

Submitted-by: t19@nikhefh.hep.nl (Geert J v Oldenborgh)
Posting-number: Volume 2, Issue 36
Archive-name: nutshell

[Documentation posted with the binaries. -sg]

#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      MAKEFILE
#      NUTSHELL.C
#
sed 's/^X//' << 'SHAR_EOF' > MAKEFILE &&
Xnutshell.prg: nutshell.o makefile
X    cc -LS=1024 -o nutshell.prg nutshell.o -lstd -ltos -lgem
X
Xnutshell.acc: nutshell.prg
X    cp -of nutshell.prg nutshell.acc
X
Xinstall: nutshell.acc
X    cp -of nutshell.acc d:/
X    cp -of nutshell.prg d:/intercom/uniterm
SHAR_EOF
chmod 0600 MAKEFILE || echo "restore of MAKEFILE fails"
sed 's/^X//' << 'SHAR_EOF' > NUTSHELL.C &&
X/*********************************************
X*                                           *
X*   NutShell, an accessory for mouse-haters.  *
```

```
X*                                                      *
X*   written by                                         *
X*        Geert Jan van Oldenborgh,                  *
X*        Ank van der Moerstraat 24,                 *
X*        NL-2331 HS Leiden,                         *
X*        tel 071-317512.                                *
X*        t19@nikhefh.hep.nl                         *
X*                                                      *
X*   and free for anyone.                            *
X*                                                      *
X************************************************/
X/*  #[ header:                                          */
X#include <stdio.h>
X#include <stdlib.h>
X#include <aes.h>
X#include <tos.h>
X#include <string.h>
X#include <ext.h>
X
X/* REDRAW: no separate screen, save only menu bar,
Xno REDRAW: save whole screen. */
X#define  REDRAW
X
X#define MOUSE_ON  graf_mouse( M_ON, (void *)0 )
X#define MOUSE_OFF graf_mouse( M_OFF, (void *)0 )
X
X#define BYTE unsigned char
X
X#define VT52_LEFT     'D'
X#define VT52_RIGHT         'C'
X#define VT52_CLRLINE  'K'
X#define VT52_CUR_ON   'e'
X#define VT52_CUR_OFF  'f'
X#define VT52_HOME     'H'
X#define VT52_CLEAR         "E<-H"[0]
X#define CRETURN '\x0D'
X
X#define PROMPT        "$ "
X
X/* prototypes */
X
Xint getscreen(void);
Xvoid redrawscreen(void);
Xvoid eventloop(void);
Xvoid putstring(char *string);
Xvoid getstring(BYTE *string, int length);
Xvoid putletter(char letter);
Xvoid putescape(char letter);
Xvoid clearcursor(void);
Xvoid newline(void);
Xvoid backspace(int n);
Xvoid putenvparent(char *entry, char *env);
Xvoid setshell(void);
```

```
Xint mysystem(char *string);
Xvoid zero_shell_p(void);
X
X/*  #] header:                                              */
X/*  #[ globals:                                        */
X
X/* AES garbage */
Xint appl_id;
Xint menu_id;
Xint buffer[16];
X
Xextern int _app;
X
X/* Basepage */
Xextern BASPAG *_BasPag;
X
X/* screen info */
X#ifndef REDRAW
Xlong    screensize = 5 + 32256L;
X#else
Xint     screensize;
X#endif
Xint     res,screenh,screenw;
Xvoid    *oldLogbase, *oldPhysbase, *newBase;
Xchar    *screen_mem, *extra_mem,
X        motd[] = "NutShell version 0.1, "\
X                 __DATE__\
X                 ", use 'lo' to exit",
X        shell[] = "d:\\shell.prg\0<- your shell",
X        home[]  = "HOME=e:\\\0<- your home",
X        error[] = "cannot execute or find shell",
X        magic[] = "GJvO",
X        st[] = "unset tosonly";
Xlong    extra_size = 0L;
Xint     extra_allocated = 0,
X        screen_allocated = 0,
X        busy = 0,
X        no_shell;
XBYTE    string[79],
X        oudstring[78] = "lo";
X
X/*  #] globals:                                        */
X/*  #[ main:                                           */
X
Xint
Xmain(void)
X/*********************************************************
X*
         *
X*   NutShell: passes commends to any running shell     *
X*   which accepts system() calls over _shell_p
     *
X*
```

```
         *
X*******************************************************/
X{
X    /* go through the all the formalities */
X
X    if ( !_app )
X    {
X         appl_id = appl_init();
X         if ( appl_id == -1 ) return(0);
X         menu_id = menu_register( appl_id, "  NutShell");
X         if ( menu_id == -1 ) return(-1);
X
X#ifdef REDRAW
X         /*   find ot the resolution, set some globals accordingly (will not
X              work on a 19" screen) */
X         res = Getrez();
X         switch ( res )
X         {
X         case 2:
X              screenh = 400;
X              screenw = 640;
X              screensize = 19*80 + 5;
X              break;
X         case 1:
X              screenh = 400;
X              screenw = 320;
X              screensize = 19*40 + 5;
X              break;
X         case 0:
X              screenh = 200;
X              screenw = 320;
X              screensize = 10*40 + 5;
X              break;
X         }
X#endif
X
X         /*   if an accessory, hit _shell_p to 0 (there cannot be a shell) */
X         Supexec(zero_shell_p);
X    }
X
X    /* and GO */
X    eventloop();
X
X    return(0);
X}
X
X/*  #] main:                                          */
X/*  #[ eventloop:                                     */
Xvoid
Xeventloop(void)
X{
X/*  #[       loop:                                        */
X    /*   the main (unending) loop: if an accesory, wait for an AC_OPEN or
```

```
X         AC_CLOSE */
X
X    while ( 1 ) {
X
X         if ( !_app ) evnt_mesag(buffer);
X         if ( busy ) continue;
X
X/*  #]       loop:                                        */
X/*  #[       open:                                        */
X         if ( _app || buffer[0] == AC_OPEN && buffer[4] == menu_id )
X         {
X              /* set a flag to indicate that we are being called
X              recursively */
X              busy = 1;
X
X              /*   The first thing to do is to free the memory we stole
X                   during startop of the .prg program (or desktop) */
X              if ( extra_allocated )
X              {
X                   extra_allocated = 0;
X                   if ( ! strcmp(extra_mem,magic) )
X                   {
X                        #ifdef DEBUG
X                        puts("freed extra_mem ");
X                        #endif
X                        Mfree( extra_mem );
X                   }
X                   else
X                        #ifdef DEBUG
X                        puts("lost extra_mem ");
X                        #endif
X                        ;
X              }
X              /* check whether we still own the screen */
X              if ( screen_allocated )
X              {
X                   if ( strcmp(screen_mem,magic) )
X                   {
X                        #ifdef DEBUG
X                        puts("lost screen ");
X                        #endif
X                        screen_allocated = 0;
X                   }
X                   else
X                   {
X                        #ifdef DEBUG
X                        puts("screen still OK ");
X                        #endif
X                   }
X              }
X              if ( getscreen() ) continue;
X
X              if ( system("free") )
```

```
X               {
X                    no_shell = 1;
X                    if ( mysystem("free") )
X                    {
X                         putstring(error);
X                         newline();
X                    }
X               }
X              else
X               no_shell = 0;
X               /* make doubly sure we only execute TOS progs (not much use)*/
X               if ( !_app ) system(st+2);
X
X               while ( 1 )
X               {
X                    putstring(PROMPT);
X                    clearcursor();
X                    getstring(string,78);
X                    if ( ! strcmp(string,"lo") ) break;
X                    else if ( ! strncmp(string,"malloc",6) )
X                    {
X                         if ( (int)strlen(string) <= 7 )
X                              putstring("Usage: malloc extra_size[kB]");
X                         else
X                         {
X                              extra_size = 1024L * atol( string + 6 );
X                              if ( extra_size )
X                                   putstring("trying");
X                              else
X                                   putstring("given");
X                         }
X                         newline();
X                    }
X                    else
X                    {
X                         if ( no_shell )
X                              mysystem(string);
X                         else
X                              system(string);
X                    }
X               }
X
X               /* clean up */
X               if ( _app )
X               {
X                    MOUSE_ON;
X                    putescape(VT52_CUR_OFF);
X                    break;
X               }
X
X               system(st);
X               redrawscreen();
X
```

```
X             if ( extra_size )
X             {
X                  if ( ( extra_mem = Malloc( extra_size ) ) > NULL )
X                  {
X                       #ifdef DEBUG
X                       puts("allocated extra_mem ");
X                       #endif
X                       strcpy(extra_mem,magic);
X                       extra_allocated = 1;
X                  }
X             }
X             else
X             {
X                  /* give up everything */
X                  #ifdef DEBUG
X                  puts("freed screen ");
X                  #endif
X                  Mfree( screen_mem );
X                  screen_allocated = 0;
X             }
X
X             busy = 0;
X        }
X/*  #]       open:                                          */
X/*  #[       close:                                         */
X        if ( buffer[0] == AC_CLOSE )
X        {
X             /*   Just for safety if we happen to receive an AC_CLOSE within one
X                  application.  Note that we rely heavily on the zeroing of
X                  memory when an application starts */
X
X                  if ( screen_allocated && ! strcmp(screen_mem,magic) )
X                  {
X                       #ifdef DEBUG
X                       puts("freed screen ");
X                       #endif
X                       Mfree(screen_mem);
X                  }
X                  if ( extra_allocated && ! strcmp(extra_mem,magic) )
X                  {
X                       #ifdef DEBUG
X                       puts("freed extra_mem ");
X                       #endif
X                       Mfree(extra_mem);
X                  }
X                  extra_allocated = 0;
X                  screen_allocated = 0;
X
X             /*   This may look strange, but remember that when running a shell
X                  the AC_CLOSE does not arrive until we start a GEM application
X                  which does an appl_init() or evnt_time(0,0).  The result is
X                  that we steal (extra_size + screensize) bytes from every .prg
X                  application, but NOT from .tos or .ttp jobs like TeX and tcc.
```

```
X             */
X             /*   I should check that I am not in the desktop, does anyone know
X                  a trick on how to do this??? */
X
X             if ( extra_size )
X             {
X                  if ( ( screen_mem = Malloc( screensize )) > NULL )
X                  {
X                       #ifdef DEBUG
X                       puts("allocated screen ");
X                       #endif
X                       screen_allocated = 2;
X                       strcpy(screen_mem,magic);
X                       if ( ( extra_mem = Malloc( extra_size )) > NULL )
X                       {
X                            #ifdef DEBUG
X                            puts("allocated extra ");
X                            #endif
X                            extra_allocated = 1;
X                            strcpy(extra_mem,magic);
X                       }
X                  }
X             }
X        }
X   }
X/*  #]       close:                                      */
X}
X/*  #] eventloop:                                        */
X/*  #[ getscreen:                                       */
X
Xint
Xgetscreen(void)
X{
X   if ( !_app )
X   {
X        if ( !screen_allocated )
X        {
X             screen_mem = Malloc( screensize + 17000L );
X             if ( screen_mem <= NULL )
X             {
X                  form_alert(1,"[3][| Not enough | memory left ][ Back to
GEM ]");
X                  busy = 0;
X                  return(1);
X             }
X             Mshrink( 0, screen_mem, screensize );
X             #ifdef DEBUG
X             puts("Allocated screen");
X             #endif
X        }
X        oldPhysbase = Physbase();
X        oldLogbase  = Logbase();
X#ifdef REDRAW
```

```
X         /*   copy the menubar to own memory */
X         memcpy(screen_mem + 5, oldPhysbase, screensize - 5);
X#else
X         /*  switch to new screen */
X         newBase = (void *)( (long)(screen_mem + 260) & 0xFFFFFF00L);
X         Setscreen( newBase, newBase, -1 );
X         Vsync();
X#endif
X    }
X    /* cursor on and clear screen */
X    MOUSE_OFF;
X    putescape(VT52_CUR_ON);
X    putescape(VT52_HOME);
X    if ( screen_allocated != 1 )
X    {
X#ifndef REDRAW
X         putescape(VT52_CLEAR);
X#endif
X         putstring(motd);
X         newline();
X         if ( !_app )
X         {
X              screen_allocated = 1;
X              strcpy(screen_mem,magic);
X         }
X    }
X    return(0);
X}
X/*  #] getscreen:                                        */
X/*  #[ redrawscreen:                                */
X
Xvoid
Xredrawscreen(void)
X{
X    /* cursor off */
X    putescape(VT52_CUR_OFF);
X#ifdef REDRAW
X    /*   copy the menubar back */
X    memcpy(oldPhysbase, screen_mem + 5, screensize - 5);
X    /*  send a redraw to all windows */
X    form_dial(FMD_FINISH, 0, 0, 0, 0, 0, 0, screenw, screenh);
X#else
X    /* set back the old screen */
X    Setscreen( oldLogbase, oldPhysbase, -1 );
X    Vsync();
X#endif
X    MOUSE_ON;
X}
X
X/*  #] redrawscreen:                                */
X/*  #[ getstring:                                   */
X
Xvoid getstring(BYTE *string, int length)
```

```
X{
X    BYTE *current, *end;
X    long key;
X    BYTE ascii, scan;
X    int i,echo;
X    char let10[] = "1234567890";
X
X    current = end = string;
X    *current = '\0';
X
X    while ( 1 )
X    {
X         /*   first use the GemDos buffer (we use the Bios, but a repeat or
X         typeahead may have filled this).  Hope nobody presses ^c in this time.
X         */
X         if ( Cconis() )
X         {
X              key = Cconin();
X              if ( key == 0L ) key = 0x00610000L; /*undo*/
X              echo = 0;
X         }
X         else
X         {
X              key = Bconin(2);
X              echo = 1;
X         }
X         ascii = (char)(key & 0x000000FFl);
X         scan = (char)( ( key >> 16 ) & 0x000000FFl);
X         if ( ascii == '\x00' )
X         {
X              switch ( scan )
X              {
X              case 0x61:
X              case 0x5d:
X                   strcpy(string,"lo");
X                   return;
X              case 0x48:
X                   strcpy(string,oudstring);
X                   backspace((int)(current-string));
X                   if ( string != end ) putescape(VT52_CLRLINE);
X                   putstring(string);
X                   current = end = string + (int)strlen(string);
X                   break;
X              case 0x50:
X                   if ( string != end )
X                   {
X                        backspace((int)(current-string));
X                        putescape(VT52_CLRLINE);
X                        current = end = string;
X                        *current = '\0';
X                   }
X                   break;
X              case 0x4b:
```

```
X                    if ( current > string )
X                    {
X                         --current;
X                         putletter('\b');
X                    }
X                    break;
X               case 0x4d:
X                    if ( current < end )
X                    {
X                         current++;
X                         putescape(VT52_RIGHT);
X                    }
X                    break;
X               case 0x47:
X                    if ( current == string )
X                    {
X                         strcpy(string,"clear");
X                         return;
X                    }
X               case 0x62:
X                    if ( current == string )
X                    {
X                         putstring(motd);
X                         newline();
X                         putstring(PROMPT);
X                    }
X                    break;
X               }
X               if ( current == string && scan >= 0x3b && scan < 0x44 )
X               {
X                    strcpy(string,"if($?PF1 )echo $PF1 ;$PF1 ");
X                    string[7] = string[18] = string[24] = let10[scan - 0x3b];
X                    return;
X               }
X               if ( current == string && scan >= 0x54 && scan < 0x5d )
X               {
X                    strcpy(string,"if($?PF1 )echo $PF1 ;$PF1 ");
X                    string[8] = string[19] = string[25] = let10[scan - 0x3b];
X                    return;
X               }
X          }
X          else
X          {
X               switch( scan )
X               {
X               case 0x4b:
X                    if ( !echo ) putletter('\b');
X                    backspace((int)(current-string));
X                    current = string;
X                    continue;
X               case 0x4d:
X                    if ( !echo ) putletter('\b');
X                    for ( i=(int)(end-current); i; --i) putescape(VT52_RIGHT);
```

```
X                current = end;
X                continue;
X            }
X            switch ( ascii )
X            {
X            case '\b':
X                if ( current == string ) break;
X                if ( echo ) putescape(VT52_LEFT);
X                --current;
X            case '\x7f':
X                if ( current == end ) break;
X                memcpy(current, current+1, (int)(end-current));
X                putstring(current);
X                putletter(' ');
X                backspace((int)(--end - current + 1));
X                break;
X            case '\x0d':
X                if ( scan == 0x72 ) current = end;
X                *current = '\0';
X                if ( current != end ) putescape(VT52_CLRLINE);
X                strcpy(oudstring,string);
X                newline();
X                return;
X            case '\x15':
X                backspace((int)(current-string));
X                putescape(VT52_CLRLINE);
X                current = end = string;
X                *current = '\0';
X                break;
X            case '\x14':
X                if ( current - string > 1 )
X                {
X                    scan = *(current-2);
X                    *(current - 2) = *(current - 1);
X                    *(current - 1) = scan;
X                    backspace(2);
X                    putstring(current - 2);
X                    backspace(end-current);
X                }
X                break;
X            default:
X                if ( (int)(end-string) < length-2 )
X                {
X                    memcpy(current + 1, current, (int)(end-current)+1);
X                    *current = ascii;
X                    end++;
X                    if ( echo ) putstring(current);
X                    backspace((int)(end - ++current));
X                }
X                else
X                    putletter('\a');
X            }
X        }
```

```
X        if ( current == end ) clearcursor();
X    }
X}
X/*  #] getstring:                                      */
X/*  #[ putthings:                                      */
X
Xvoid
Xputstring(char *string)
X{
X    char *p;
X    p = string;
X    while ( *p ) Bconout(5,(int)(*p++));
X}
Xvoid
Xputletter(char letter)
X{
X    Bconout(2,(int)letter);
X}
Xvoid
Xputescape(char letter)
X{
X    Bconout(2,0x1b);
X    Bconout(2,(int)letter);
X}
Xvoid
Xclearcursor(void)
X{
X    putletter(' ');
X    putescape(VT52_LEFT);
X}
Xvoid
Xnewline(void)
X{
X    putletter('\x0d');
X    putletter('\n');
X}
Xvoid
Xbackspace(int n)
X{
X    int i;
X    for( i=n; i; --i) putescape(VT52_LEFT);
X}
X/*  #] putthings:                                      */
X/*  #[ mysystem:                                       */
Xint mysystem(char *string)
X{
X    char arg[82],env[200],*p,*q;
X    long l;
X    int i,j;
X
X    strcpy(arg,"x-c ");
X    strcat(arg,string);
X    arg[0] = strlen(arg+1);
```

```
X#ifdef DEBUG
X    printf("mysytem calls Pexec with argument %s\n",arg+1);
X#endif
X    p = env;
X    q = "unixpath=";
X    while ( *p++ = *q++ );
X    q = home;
X    while ( *p++ = *q++ );
X    q = "_PBP=";
X    while ( *p++ = *q++ );
X    l = (long)_BasPag;
X    p += 7;
X    for( i=0; i<8; i++)
X    {
X         j = l & 0x0000000FL;
X         *--p = "0123456789ABCDEF"[j];
X         l >>= 4;
X    }
X    p += 8;
X    *p++ = 0;
X    q = "ARGV=CCC????????????????????????";
X    while ( *p++ = *q++ );
X    q = shell;
X    while ( *p++ = *q++ );
X    q = "-c";
X    while ( *p++ = *q++ );
X    q = string;
X    while ( *p++ = *q++ );
X    *p = 0;
X#ifdef DEBUG
X    p = env;
X    putstring("environment:");
X    newline();
X    while (*p)
X    {
X         while(*p++) putletter(*p);
X         newline();
X    }
X#endif
X    l = Pexec(0, shell, arg, env);
X    putescape(VT52_CUR_ON);
X    return( (int)l);
X}
X/*  #] mysystem:                                    */
X/*  #[ zero_shell_p:                                */
X    void zero_shell_p(void)
X    {
X         long *p;
X         p = 0x4f6;
X         *p = 0L;
X    }
X/*  #] zero_shell_p:                                */
X
```

```
SHAR_EOF
chmod 0600 NUTSHELL.C || echo "restore of NUTSHELL.C fails"
exit 0


From 5417562290147725951
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,4541c49b194ea09b,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-04-14 00:20:52 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
agate!saturn!ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i037: texchang -- Change files for BibTeX, Tangle, and Weave
Keywords: shar
Message-ID: <6881@saturn.ucsc.edu>
Date: 14 Apr 89 07:20:52 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1008
Approved: koreth@ssyx.ucsc.edu

Submitted-by: csrobe@cs.wm.edu (Chip Roberson)
Posting-number: Volume 2, Issue 37
Archive-name: texchang

[I'm still not too sure why this is needed, but what the heck. -sg]

The TeX system comes with a 'relatively' generic base of programs written
in Web that operate on some real or imaginary computer.  To compile these
programs for your system, you find an appropriate 'changefile' that describes
the system dependecies for your machine and include it into make.  This is
just the way god-emperor Knuth decided to do it.  It truly is a tangled
web that he has weaved.  ;-)

Anyway, anyone who wants to build a version of BibTeX, Tangle, and Weave
can get the original from the standard TeX archives at score and then
include their changefile and compile with their favorite Pascal compiler.
OR, they can wait until someone does the web2c conversion and creates
the appropriate changes there.  This seemed to be the quickest way
to get these programs to those who needed them.

#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      bibtex
#      tangle
#      weave
#
sed 's/^X//' << 'SHAR_EOF' > bibtex &&
XBibTeX change file for the Atari ST
XAuthor: Nico Poppelier.
```

```
XDate  : December 27, 1988.
X
XSince the Prospero Pascal compiler strictly adheres to the
XISO Pascal standard (level 0), all occurrences of ':0' or ': 0'
Xin write and writeln statements have to be replaced by
X':1' and ': 1', respectively.
X(This was reported to Oren Patashnik, see TeXhax of Dec 88).
X
X@x
X@d banner=='This is BibTeX, Version 0.99c' {printed when the program starts}
X@y
X@d banner=='This is BibTeX, Version 0.99c for the Atari ST' {printed when the pr
X   ogram starts}
X@z
X
X@x
X@d term_out == tty
X@d term_in == tty
X@y
XFiles 'input' and 'output' are used for terminal input and output.
XProspero Pascal for the Atari ST requires us to mention
X|input| and |output| in the program header.
X@d term_out == output
X@d term_in == input
X@z
X
X@x
X@d debug == @{          { remove the `|@{|' when debugging }
X@d gubed == @t@>@}      { remove the `|@}|' when debugging }
X@y
X@d debug ==             { we are debugging }
X@d gubed == @t@>
X@z
X
X@x
X@d stat == @{           { remove the `|@{|' when keeping statistics }
X@d tats == @t@>@}       { remove the `|@}|' when keeping statistics }
X@y
X@d stat ==              { we are keeping statistics }
X@d tats == @t@>
X@z
X
X@x
X@d othercases == others: {default for cases not listed explicitly}
X@y
X@d othercases == otherwise {default for cases not listed explicitly}
X@z
X
X@x
Xprogram BibTEX;                {all files are opened dynamically}
X@y
Xprogram BibTEX(@!input,@!output); {these files have to be specified here}
X@z
```

```
X
X@x
X@<Compiler directives@>=
X@{@&$C-,A+,D-@}  {no range check, catch arithmetic overflow, no debug overhead}
X@!debug @{@&$C+,D+@}@+ gubed  {but turn everything on when debugging}
X@y
XOn the Atari ST, things are different: Prospero Pascal does
Xnot allow in-comment directives
X@<Compiler directives@>=
X@z
X
X@x
X    write_ln (term_out,bad:0,' is a bad bad');
X@y
X    write_ln (term_out,bad:1,' is a bad bad');
X@z
X
X@x
X@!alpha_file=packed file of text_char;  {files that contain textual data}
X@y
X@!alpha_file=text;      {files that contain textual data}
X@z
X
X@x
X@d reset_OK(#)==erstat(#)=0
X@d rewrite_OK(#)==erstat(#)=0
X@y
X@d reset_OK(#)==fstat(#)
X@d rewrite_OK(#)==fstat(#)
X@z
X
X@x
Xfunction erstat(var f:file):integer; extern; {in the runtime library}
X@y
X@z
X
X@x
Xbegin reset(f,name_of_file,'/O'); a_open_in:=reset_OK(f);
X@y
Xbegin assign(f,name_of_file);
X      reset(f); a_open_in:=reset_OK(name_of_file);
X@z
X
X@x
Xbegin rewrite(f,name_of_file,'/O'); a_open_out:=rewrite_OK(f);
X@y
Xbegin assign(f,name_of_file);
X      rewrite(f); a_open_out:=rewrite_OK(name_of_file);
X@z
X
X@x
Xprocedure a_close(var f:alpha_file);  {close a text file}
Xbegin close(f);
```

```
X@y
Xprocedure a_close(var f:alpha_file);  {close a text file}
Xbegin close(f, true);
X@z
X
X@x
X@d overflow(#)==begin           {fatal error---close up shop}
X                print_overflow;
X                print_ln(#:0);
X@y
X@d overflow(#)==begin           {fatal error---close up shop}
X                print_overflow;
X                print_ln(#:1);
X@z
X
X@x
X    confusion ('Illegal string number:',s:0);
X@y
X    confusion ('Illegal string number:',s:1);
X@z
X
X@x
Xprint ('---line ',cur_aux_line:0,' of file ');
X@y
Xprint ('---line ',cur_aux_line:1,' of file ');
X@z
X
X@x
Xprint ('A level-',aux_ptr:0,' auxiliary file: ');
X@y
Xprint ('A level-',aux_ptr:1,' auxiliary file: ');
X@z
X
X@x
Xprint ('--line ',bst_line_num:0,' of file ');
X@y
Xprint ('--line ',bst_line_num:1,' of file ');
X@z
X
X@x
X  trace_pr_ln (' is an integer literal with value ',token_value:0);
X@y
X  trace_pr_ln (' is an integer literal with value ',token_value:1);
X@z
X
X@x
X    print (single_ptr + wiz_def_ptr : 0,': ');
X@y
X    print (single_ptr + wiz_def_ptr : 1,': ');
X@z
X
X@x
Xprint ('--line ',bib_line_num:0,' of file ');
```

```
X@y
Xprint ('--line ',bib_line_num:1,' of file ');
X@z
X
X@x
X    print ('Database file #',bib_ptr+1:0,': ');
X@y
X    print ('Database file #',bib_ptr+1:1,': ');
X@z
X
X@x
X    print_ln (total_fields:0,' fields:');
X@y
X    print_ln (total_fields:1,' fields:');
X@z
X
X@x
Xbib_err ('Your field is more than ',buf_size:0,' characters');
X@y
Xbib_err ('Your field is more than ',buf_size:1,' characters');
X@z
X
X@x
X    bib_err ('You''ve exceeded ',max_bib_files:0,' preamble commands');
X@y
X    bib_err ('You''ve exceeded ',max_bib_files:1,' preamble commands');
X@z
X
X@x
X  trace_pr_ln ('Sorting ',left_end:0,' through ',right_end:0);
X@y
X  trace_pr_ln ('Sorting ',left_end:1,' through ',right_end:1);
X@z
X
X@x
X    stk_int : trace_pr_ln (lit_stack[lit_stk_ptr]:0);
X@y
X    stk_int : trace_pr_ln (lit_stack[lit_stk_ptr]:1);
X@z
X
X@x
X    stk_int : print (stk_lt:0,' is an integer literal');
X@y
X    stk_int : print (stk_lt:1,' is an integer literal');
X@z
X
X@x
X    stk_int : print_ln (stk_lt:0);
X@y
X    stk_int : print_ln (stk_lt:1);
X@z
X
X@x
```

```
X    print_ln ('ptr=',lit_stk_ptr:0,', stack=');
X@y
X    print_ln ('ptr=',lit_stk_ptr:1,', stack=');
X@z
X
X@x
X      print_ln ('Pointer is ',str_ptr:0,' but should be ',cmd_str_ptr:0);
X@y
X      print_ln ('Pointer is ',str_ptr:1,' but should be ',cmd_str_ptr:1);
X@z
X
X@x
X        bst_string_size_exceeded (ent_str_size:0,', the entry');
X@y
X        bst_string_size_exceeded (ent_str_size:1,', the entry');
X@z
X
X@x
X            bst_string_size_exceeded (glob_str_size:0,', the global');
X@y
X            bst_string_size_exceeded (glob_str_size:1,', the global');
X@z
X
X@x
X        print ('There aren''t ',pop_lit2:0,' names in "');
X@y
X        print ('There aren''t ',pop_lit2:1,' names in "');
X@z
X
X@x
X                print ('Name ',pop_lit2:0,' in "');
X@y
X                print ('Name ',pop_lit2:1,' in "');
X@z
X
X@x
X    print ('Too many commas in name ',pop_lit2:0,' of "');
X@y
X    print ('Too many commas in name ',pop_lit2:1,' of "');
X@z
X
X@x
Xprint ('Name ',pop_lit2:0,' of "');
X@y
Xprint ('Name ',pop_lit2:1,' of "');
X@z
X
X@x
X    bst_ex_warn (pop_lit1:0,' isn''t valid ASCII');
X@y
X    bst_ex_warn (pop_lit1:1,' isn''t valid ASCII');
X@z
X
```

```
X@x
X    print ('Aborted at line ',bib_line_num:0,' of file ');
X@y
X    print ('Aborted at line ',bib_line_num:1,' of file ');
X@z
X
X@x
X    trace_pr_ln ('The ',num_bib_files:0,' database files are');
X@y
X    trace_pr_ln ('The ',num_bib_files:1,' database files are');
X@z
X
X@x
X    trace_pr ('all_marker=',all_marker:0,', ');
X@y
X    trace_pr ('all_marker=',all_marker:1,', ');
X@z
X
X@x
X    trace_pr_ln ('old_num_cites=',old_num_cites:0)
X@y
X    trace_pr_ln ('old_num_cites=',old_num_cites:1)
X@z
X
X@x
Xtrace_pr ('The ',num_cites:0);
X@y
Xtrace_pr ('The ',num_cites:1);
X@z
X
X@x
Xif (type_list[cite_ptr] = undefined) then
X    undefined : trace_pr ('unknown')
X@y
Xif (type_list[cite_ptr] = undefined) then
X    trace_pr ('unknown')
X@z
X
X@x
X        trace_pr (' ',entry_ints[int_ent_ptr]:0);
X@y
X        trace_pr (' ',entry_ints[int_ent_ptr]:1);
X@z
X
X@x
X            trace_pr_ln (wiz_fn_ptr:0,'--end-of-def--')
X@y
X            trace_pr_ln (wiz_fn_ptr:1,'--end-of-def--')
X@z
X
X@x
X            trace_pr (wiz_fn_ptr:0,'  quote_next_function    ')
X@y
```

```
X            trace_pr (wiz_fn_ptr:1,'  quote_next_function    ')
X@z
X
X@x
X            trace_pr (wiz_fn_ptr:0,'  `');
X@y
X            trace_pr (wiz_fn_ptr:1,'  `');
X@z
X
X@x
Xstat_pr ('You''ve used ',num_cites:0);
X@y
Xstat_pr ('You''ve used ',num_cites:1);
X@z
X
X@x
Xstat_pr_ln ('            ',wiz_def_ptr:0,' wiz_defined-function locations,');
X@y
Xstat_pr_ln ('            ',wiz_def_ptr:1,' wiz_defined-function locations,');
X@z
X
X@x
Xstat_pr_ln ('            ',str_ptr:0,' strings with ',str_start[str_ptr]:0,
X@y
Xstat_pr_ln ('            ',str_ptr:1,' strings with ',str_start[str_ptr]:1,
X@z
X
X@x
Xstat_pr_ln ('and the built_in function-call counts, ', total_ex_count:0,
X@y
Xstat_pr_ln ('and the built_in function-call counts, ', total_ex_count:1,
X@z
X
X@x
X    stat_pr_ln (' -- ',execution_count[blt_in_ptr]:0);
X@y
X    stat_pr_ln (' -- ',execution_count[blt_in_ptr]:1);
X@z
X
X@x
X                        print_ln ('(There were ',err_count:0,' warnings)');
X@y
X                        print_ln ('(There were ',err_count:1,' warnings)');
X@z
X
X@x
X                      print_ln ('(There were ',err_count:0,
X@y
X                      print_ln ('(There were ',err_count:1,
X@z
X
X
SHAR_EOF
```

```
chmod 0600 bibtex || echo "restore of bibtex fails"
sed 's/^X//' << 'SHAR_EOF' > tangle &&
XTANGLE change file for the Atari ST
XAuthor: Nico Poppelier.
XDate  : December 30, 1988.
X
XDue to an error in the Prospero Pascal compiler the control variable
Xof the FOR loops in sections 138, 182, and 184 are judged 'insecure',
Xi.e., it looks as if they are modified by a routine.
XTo avoid this 'error' a new loop variable k is introduced.
X
X@x
X\pageno=\contentspagenumber \advance\pageno by 1
X@y
X\pageno=\contentspagenumber \advance\pageno by 1
X% \let\maybe=\iffalse
X% \def\title{TANGLE changes for the Atari ST}
X\def\title{TANGLE, Version 2.8 for the Atari ST}
X@z
X
X@x
X@d banner=='This is TANGLE, Version 2.8'
X@y
X@d banner=='This is TANGLE, Version 2.8 for the Atari ST'
X@z
X
X@x
Xand the string pool output goes to file |pool|.
X@y
Xand the string pool output goes to file |pool|.
XProspero Pascal for the Atari ST requires us to mention
X|input| and |output| in the program header, too.
XThey are used for terminal input and output.
X@z
X
X@x
Xprogram TANGLE(@!web_file,@!change_file,@!Pascal_file,@!pool);
X@y
Xprogram TANGLE(@!input,@!output,
X               @!web_file,@!change_file,@!Pascal_file,@!pool);
X@z
X
X@x
Xprocedure initialize;
X@y
Xprocedure wait(delay:real); external; {from doslib}@/
Xprocedure initialize;
X@z
X
X@x
X@d debug==@{ {change this to `$\\{debug}\equiv\null$' when debugging}
X@d gubed==@t@>@} {change this to `$\\{gubed}\equiv\null$' when debugging}
X@y
```

```
X@d debug== {we are debugging}
X@d gubed==
X@z
X
X@x
X@d stat==@{ {change this to `$\\{stat}\equiv\null$'
X  when gathering usage statistics}
X@d tats==@t@>@} {change this to `$\\{tats}\equiv\null$'
X  when gathering usage statistics}
X@y
X@d stat== {we are gathering usage statistics}
X@d tats==
X@z
X
X@x
X@<Compiler directives@>=
X@{@&$C-,A+,D-@} {no range check, catch arithmetic overflow, no debug overhead}
X@!debug @{@&$C+,D+@}@+ gubed {but turn everything on when debugging}
X@y
XOn the Atari ST, things are different:
XProspero Pascal does not allow in-comment directives
X@<Compiler directives@>=
X@z
X
X@x
X@d othercases == others: {default for cases not listed explicitly}
X@y
X@d othercases == otherwise {default for cases not listed explicitly}
X@z
X
X@x
X@!max_toks=50000; {|1/zz| times the number of bytes in compressed \PASCAL\ code;
X
X  must be less than 65536}
X@y
X@!max_toks=51000; {|1/zz| times the number of bytes in compressed \PASCAL\ code;
X
X  must be less than 65536}
X@z
X
X@x
X@!text_file=packed file of text_char;
X@y
X@!text_file=text;
X@z
X
X@x
X@d print(#)==write(term_out,#) {`|print|' means write on the terminal}
X@y
X@d term_in==input
X@d term_out==output
X@d print(#)==write(term_out,#) {`|print|' means write on the terminal}
X@z
```

```
X
X@x
X@<Globals...@>=
X@!term_out:text_file; {the terminal as an output file}
X@y
X@z
X
X@x
Xrewrite(term_out,'TTY:'); {send |term_out| output to the terminal}
X@y
Xprint_ln(banner); {print a ``banner line''}
X@z
X
X@x
X@d update_terminal == break(term_out) {empty the terminal output buffer}
X@y
X@d update_terminal == {emptying the screen output buffer is not necessary}
X@z
X
X@x
X@ The following code opens |Pascal_file| and |pool|.
XSince these files were listed in the program header, we assume that the
X\PASCAL\ runtime system has checked that suitable external file names have
Xbeen given.
X@^system dependencies@>
X
X@<Set init...@>=
Xrewrite(Pascal_file); rewrite(pool);
X@y
X@ The following code opens |Pascal_file| and |pool|.
XActually, on the Atari ST this task is put off until later.
X@^system dependencies@>
X@z
X
X@x
X@!string_ptr:name_pointer; {next number to be given to a string of length |<>1|}
X@y
X@!string_ptr:name_pointer; {next number to be given to a string of length |<>1|}
X@!k:integer; {extra loop variable}
X@z
X
X@x
X  begin for loc:=0 to change_limit do buffer[loc]:=change_buffer[loc];
X@y
X  begin for k:=0 to change_limit do buffer[k]:=change_buffer[k];
X@z
X
X@x
X@!term_in:text_file; {the user's terminal as an input file}
X@y
X@z
X
X@x
```

```
Xreset(term_in,'TTY:','/I'); {open |term_in| as the terminal, don't do a |get|}
X@y
X@z
X
X@x
Xprint_ln(banner); {print a ``banner line''}
X@y
X@z
X
X@x
Xstat for zo:=0 to zz-1 do max_tok_ptr[zo]:=tok_ptr[zo];@+tats@;@/
X@y
Xstat for k:=0 to zz-1 do max_tok_ptr[k]:=tok_ptr[k];@+tats@;@/
X@z
X
X@x save pool and Pascal files only if they were written to.
Xif string_ptr>128 then @<Finish off the string pool file@>;
Xstat @<Print statistics about memory usage@>;@+tats@;@/
X@t\4\4@>{here files should be closed if the operating system requires it}
X@<Print the job |history|@>;
Xend.
X@y
Xif history<fatal_message then begin
X  if string_ptr>128 then begin @<Finish off the string pool file@>;
X    close(pool, true);
X  end;
X  close(Pascal_file, true);
Xend;
Xstat @<Print statistics about memory usage@>;@+tats@;@/
X@<Print the job |history|@>;
Xwait(3);
Xend.
X@z
X
X@x
Xfor string_ptr:=1 to 9 do
X@y
Xfor k:=1 to 9 do
X@z
X
X@x
Xfor string_ptr:=9 downto 1 do write(pool,xchr["0"+out_buf[string_ptr]]);
X@y
Xfor k:=9 downto 1 do write(pool,xchr["0"+out_buf[k]]);
X@z
X
X@x
Xfor zo:=1 to zz-1 do print('+',max_tok_ptr[zo]:1);
X@y
Xfor k:=1 to zz-1 do print('+',max_tok_ptr[k]:1);
X@z
X
X@x
```

```
XThis module should be replaced, if necessary, by changes to the program
Xthat are necessary to make \.{TANGLE} work at a particular installation.
XIt is usually best to design your change file so that all changes to
Xprevious modules preserve the module numbering; then everybody's version
Xwill be consistent with the printed program. More extensive changes,
Xwhich introduce new modules, can be inserted here; then only the index
Xitself will get a new module number.
X@y
XHere are the remaining changes to the program
Xthat are necessary to make \.{TANGLE} work on the Atari ST.
X
X@ @<Local...@>=
X@!ask,@!name_found: boolean;
X@!file_name,@!job_name: string;
X@!name_length,@!name_index: integer;
X
X@ Get the external file names, and then call |open|
Xto associate an external file with each file variable.
X
X@<Set init...@>=
Xgetcomm(job_name);
Xname_length:=l @& e @& n @& g @& t @& h(job_name); name_index:=1;
Xwhile (name_index <= name_length) and
X      (job_name[name_index] = ' ') do incr(name_index);
Xname_found:=name_index <= name_length;
Xif name_found then delete(job_name, 1, name_index-1);
X
Xif name_found then begin
X  file_name:=concat(job_name,'.WEB');
X  assign(web_file,file_name);
X  ask:=not fstat(file_name);
X  if ask then write_ln('Couldn''t open ',file_name);
Xend
Xelse ask:=true;
Xwhile ask do begin
X  write('Web file: ');
X  if eof then begin mark_fatal; jump_out; end;
X  read_ln(file_name);
X  if checkfn(file_name) then begin
X    ask:= not fstat(file_name);
X    if ask then write_ln('Couldn''t open ',file_name)
X           else assign(web_file,file_name);
X  end
X  else begin
X    ask:=true; write_ln('"',file_name,'" is not a good filename')
X  end;
Xend;
X
Xif name_found then begin
X  file_name:=concat(job_name,'.CH');
X  assign(change_file,file_name);
X  ask:=not fstat(file_name);
X  if ask then write_ln('Couldn''t open ',file_name);
```

```
Xend
Xelse ask:=true;
Xwhile ask do begin
X  write('Change file: ');
X  if eof then begin mark_fatal; jump_out; end;
X  read_ln(file_name);
X  if file_name='' then file_name:='empty.ch';
X  if checkfn(file_name) then begin
X    ask:=not fstat(file_name);
X    if ask then write_ln('Couldn''t open ',file_name)
X           else assign(change_file,file_name);
X  end
X  else begin
X    ask:=true; write_ln('"',file_name,'" is not a good filename')
X  end;
Xend;
X
Xif name_found then begin
X  file_name:=concat(job_name,'.PAS');
X  assign(Pascal_file,file_name);
X  ask:=not checkfn(file_name);
X  if ask then write_ln('"',file_name,'" is not a good filename')
Xend
Xelse ask:=true;
Xwhile ask do begin
X  write('Pascal file: ');
X  if eof then begin mark_fatal; jump_out; end;
X  read_ln(file_name);
X  ask := not checkfn(file_name);
X  if ask then
X    write_ln('"',file_name,'" is not a good filename')
X  else
X    assign(Pascal_file,file_name);
Xend;
X
Xif name_found then begin
X  file_name:=concat(job_name,'.POO');
X  assign(pool,file_name);
X  ask:=not checkfn(file_name);
X  if ask then write_ln('"',file_name,'" is not a good filename')
Xend
Xelse ask:=true;
Xwhile ask do begin
X  write('Pool file: ');
X  if eof then begin mark_fatal; jump_out; end;
X  read_ln(file_name);
X  ask := not checkfn(file_name);
X  if ask then
X    write_ln('"',file_name,'" is not a good filename')
X  else
X    assign(pool,file_name);
Xend;
X
```

```
Xrewrite(Pascal_file); rewrite(pool);
X@z
X
X
X
SHAR_EOF
chmod 0600 tangle || echo "restore of tangle fails"
sed 's/^X//' << 'SHAR_EOF' > weave &&
XWEAVE change file for the Atari ST
XAuthor: Nico Poppelier.
XDate  : December 30, 1988.
X        Increased max_refs and max_toks as in Version 2.6.
X
XDue to an error in the Prospero Pascal compiler the control
Xvariable of the FOR loop in section 85 is judged 'insecure',
Xi.e., it looks as if it is modified by a routine.
XTo avoid this error a new loop variable k is introduced.
X
X@x
X\pageno=\contentspagenumber \advance\pageno by 1
X@y
X\pageno=\contentspagenumber \advance\pageno by 1
X% \let\maybe=\iffalse
X% \def\title{WEAVE changes for the Atari ST}
X\def\title{WEAVE, Version 2.8 for the Atari ST}
X@z
X
X@x
X@d banner=='This is WEAVE, Version 2.8'
X@y
X@d banner=='This is WEAVE, Version 2.8 for the Atari ST'
X@z
X
X@x
Xand |change_file|, and the \TeX\ output goes to file |tex_file|.
X@y
Xand |change_file|, and the \TeX\ output goes to file |tex_file|.
XProspero Pascal requires us to mention |input| and |output|
Xin the program header, too.
XThey are used for terminal input and output.
X@z
X
X@x
Xprogram WEAVE(@!web_file,@!change_file,@!tex_file);
X@y
Xprogram WEAVE(@!input,@!output,
X              @!web_file,@!change_file,@!tex_file);
X@z
X
X@x
Xprocedure initialize;
X@y
Xprocedure wait(delay:real); external;
```

```
Xprocedure initialize;
X@z
X
X@x
X  begin @<Set initial values@>@/
X@y
X  begin @<Preset initial values@>@/
X  @<Set initial values@>@/
X@z
X
X@x
X@d debug==@{ {change this to `$\\{debug}\equiv\null$' when debugging}
X@d gubed==@t@>@} {change this to `$\\{gubed}\equiv\null$' when debugging}
X@y
X@d debug== {we are debugging}
X@d gubed==
X@z
X
X@x
X@d stat==@{ {change this to `$\\{stat}\equiv\null$'
X  when gathering usage statistics}
X@d tats==@t@>@} {change this to `$\\{tats}\equiv\null$'
X  when gathering usage statistics}
X@y
X@d stat== {we are gathering usage statistics}
X@d tats==
X@z
X
X@x
X@<Compiler directives@>=
X@{@&$C-,A+,D-@} {no range check, catch arithmetic overflow, no debug overhead}
X@!debug @{@&$C+,D+@}@+ gubed {but turn everything on when debugging}
X@y
XOn the Atari ST, things are a bit different:
XProspero Pascal does not allow in-comment directives.
X
X@<Compiler directives@>=
X@z
X
X@x
X@d othercases == others: {default for cases not listed explicitly}
X@y
X@d othercases == otherwise {default for cases not listed explicitly}
X@z
X
X@x
X@!max_refs=20000; {number of cross references; must be less than 65536}
X@!max_toks=20000; {number of symbols in \PASCAL\ texts being parsed;
X@y
X@!max_refs=30000; {number of cross references; must be less than 65536}
X@!max_toks=30000; {number of symbols in \PASCAL\ texts being parsed;
X@z
X
```

```
X@x
X@!text_file=packed file of text_char;
X@y
X@!text_file=text;
X@z
X
X@x
X@d print(#)==write(term_out,#) {`|print|' means write on the terminal}
X@y
X@d term_in==input
X@d term_out==output
X@d print(#)==write(term_out,#) {`|print|' means write on the terminal}
X@z
X
X@x
X@!term_out:text_file; {the terminal as an output file}
X@y
X@z
X
X@x
Xrewrite(term_out,'TTY:'); {send |term_out| output to the terminal}
X@y
X@z
X
X@x
X@d update_terminal == break(term_out) {empty the terminal output buffer}
X@y
X@d update_terminal == {emptying the output buffer is not necessary}
X@z
X
X@x
X@ The following code opens |tex_file|.
XSince this file was listed in the program header, we assume that the
X\PASCAL\ runtime system has checked that a suitable external file name has
Xbeen given.
X@^system dependencies@>
X
X@<Set init...@>=
Xrewrite(tex_file);
X@y
X@ The following code opens |tex_file|.
XActually, on the Atari ST this task is put off until later.
X@^system dependencies@>
X@z
X
X@x
X@!loc:0..long_buf_size; {the next character position to be read from the buffer}
X@y
X@!loc:0..long_buf_size; {the next character position to be read from the buffer}
X@!k:integer; {extra loop variable}
X@z
X
X@x
```

```
X  begin for loc:=0 to change_limit do buffer[loc]:=change_buffer[loc];
X@y
X  begin for k:=0 to change_limit do buffer[k]:=change_buffer[k];
X@z
X
X@x
X@!term_in:text_file; {the user's terminal as an input file}
X@y
X@z
X
X@x
Xreset(term_in,'TTY:','/I'); {open |term_in| as the terminal, don't do a |get|}
X@y
X@z
X
X@x
Xprint_ln(banner); {print a ``banner line''}
X@y
X@z
X
X@x
X@t\4\4@>{here files should be closed if the operating system requires it}
X@<Print the job |history|@>;
Xend.
X@y
Xif history<fatal_message then
X        @= close@>(tex_file, true);
X@<Print the job |history|@>; wait(3);
Xend.
X@z
X
X@x
XThis module should be replaced, if necessary, by changes to the program
Xthat are necessary to make \.{WEAVE} work at a particular installation.
XIt is usually best to design your change file so that all changes to
Xprevious modules preserve the module numbering; then everybody's version
Xwill be consistent with the printed program. More extensive changes,
Xwhich introduce new modules, can be inserted here; then only the index
Xitself will get a new module number.
X@y
XHere are the remaining changes to the program
Xthat are necessary to make \.{WEAVE} work on the Atari ST.
X
X@ @<Local...@>=
X@!ask,@!name_found:boolean;
X@!file_name,@!job_name:s @& t @& r @& i @& n @& g;
X@!name_length,@!name_index: integer;
X
X@ Get the external file names, and then call |open|
Xto associate an external file with each file variable.
X
X@<Preset init...@>=
Xprint_ln(banner); {print a ``banner'' line}
```

```
X
Xgetcomm(job_name);
Xname_length:=l @& e @& n @& g @& t @& h(job_name); name_index:=1;
Xwhile (name_index <= name_length) and
X      (job_name[name_index] = ' ') do incr(name_index);
Xname_found:=name_index <= name_length;
Xif name_found then delete(job_name, 1, name_index-1);
X
Xif name_found then begin
X  file_name:=concat(job_name,'.WEB');
X  assign(web_file,file_name);
X  ask:=not fstat(file_name);
X  if ask then write_ln('Couldn''t open ',file_name);
Xend
Xelse ask:=true;
Xwhile ask do begin
X  write('Web file: ');
X  if eof then begin mark_fatal; jump_out; end;
X  read_ln(file_name);
X  if checkfn(file_name) then begin
X    ask:= not fstat(file_name);
X    if ask then write_ln('Couldn''t open ',file_name)
X           else assign(web_file,file_name);
X  end
X  else begin
X    ask:=true; write_ln('"',file_name,'" is not a good filename')
X  end;
Xend;
X
Xif name_found then begin
X  file_name:=concat(job_name,'.CH');
X  assign(change_file,file_name);
X  ask:=not fstat(file_name);
X  if ask then write_ln('Couldn''t open ',file_name);
Xend
Xelse ask:=true;
Xwhile ask do begin
X  write('Change file: ');
X  if eof then begin mark_fatal; jump_out; end;
X  read_ln(file_name);
X  if file_name='' then file_name:='empty.ch';
X  if checkfn(file_name) then begin
X    ask:=not fstat(file_name);
X    if ask then write_ln('Couldn''t open ',file_name)
X           else assign(change_file,file_name);
X  end
X  else begin
X    ask:=true; write_ln('"',file_name,'" is not a good filename')
X  end;
Xend;
X
Xif name_found then begin
X  file_name:=concat(job_name,'.TEX');
```

```
X  assign(tex_file,file_name);
X  ask:=not checkfn(file_name);
X  if ask then write_ln('"',file_name,'" is not a good filename')
Xend
Xelse ask:=true;
Xwhile ask do begin
X  write('TeX file: ');
X  if eof then begin mark_fatal; jump_out; end;
X  read_ln(file_name);
X  ask := not checkfn(file_name);
X  if ask then
X    write_ln('"',file_name,'" is not a good filename')
X  else
X    assign(tex_file,file_name);
Xend;
X
Xrewrite(tex_file);
X@z
X
X
SHAR_EOF
chmod 0600 weave || echo "restore of weave fails"
exit 0
```

```
From -1610824704661271
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,79674dba13de3c4d,start
X-Google-Attributes: gid100493,public
X-Google-Thread: 111a55,79674dba13de3c4d,start
X-Google-Attributes: gid111a55,public
X-Google-ArrivalTime: 1989-04-14 15:39:15 PST
Xref: utzoo comp.sources.atari.st:151 comp.sys.atari.st:13590
Path: utzoo!attcan!uunet!husc6!mailrus!csd4.milw.wisc.edu!bionet!agate!saturn!
ssyx.ucsc.edu!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st,comp.sys.atari.st
Subject: v02INF1: Introduction to comp.sources.atari.st
Message-ID: <6059@saturn.ucsc.edu>
Date: 22 Jan 89 22:44:55 GMT
Expires: 22 May 89 22:44:54 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 144
Approved: koreth@ssyx.UCSC.EDU

Submitted-by: koreth@ssyx.ucsc.edu (Steven Grimm)
Posting-number: Volume 2, Info 1
Archive-name: intro

This is the first of two introductory articles about comp.sources.atari.st.
This one describes how to submit source to the newsgroup, where the
archive sites are, and how to contact them.  The companion article lists
all previously-published sources.
```

I am always looking for suggestions on how to improve the usefulness
of the newsgroup, and can be contacted as listed below.

-- Steven Grimm

Subject: Submitting source for publication

Items intended for posting or queries and problem notes should be sent to
atari-sources@ssyx.ucsc.edu. If you are on a UUCP-only site, you can send
them to ucbvax!ssyx!atari-sources.

If you want verification of arrival, so say in a cover note, or at the
beginning of your submission, if it is small. I try to verify that a
program works, and if I can't get it to work, I may hold up posting it
for a couple of days. Please note that, except in rare cases, source
without documentation will not be published. The backlog from receipt
to posting varies from one to four weeks depending mostly on the set
of submissions currently in my queue.

I prefer that you submit sources and binaries separately, as it's a
pain to separate them, and such submissions are invariably placed on
the back burner. Sources should be submitted in "shar" format unless
you don't have access to a shar program, in which case "arc" is all
right.

Please inform me of any special system requirements, and what development
environment was used, so that I can properly test your submission.

Subject: The structure of comp.sources.atari.st articles

Each posting in comp.sources.atari.st is called an "issue"; there are
roughly 100 issues to a volume. The division is arbitrary and may vary.
There are two types of articles in comp.sources.atari.st: sources and
"information postings." They can be distinguished by the subject line:

  Subject: v01INF3: Index for Volume 1 and other info

This first word in the title identifies this as the third info posting of
volume one. Similarly, the subject line shown below:

  Subject: v01i081: tossrc -- Source code to TOS

identifies this as the 81st source article in Volume 1. Large sources are
broken up into smaller pieces, and have subject lines that look like
this:

  Subject: v01i082: VMS clone in ST-LOGO, Part03/08

Certain information about the system configuration required to use the program is given on the keywords line.

```
  Keywords: MWC, Megamax, 1meg, high
```

This means that the program compiles with Mark Williams or Megamax C, requires at least one meg of RAM and only runs in high resolution. Following is a list of keywords; new ones may be added as needed. They are mostly self-explanatory.

```
  Alcyon  - Alcyon C (included with Atari developer's kit)
  MWC         - Mark Williams C
  Megamax - Megamax C
  Laser       - Laser C
  Lattice - Lattice C
  C       - Any C compiler
  OSI         - OSI Personal Pascal
  BASIC       - ST BASIC (this is not a joke!)
  uuencode    - program is uuencoded (uudecode required to unpack)
  arc         - program is archived (arc required to unpack)
  high        - high resolution
  medium  - medium resolution
  low         - low resolution
  1meg        - needs 1 meg of RAM
```

The first few lines of an article are auxiliary headers that look like this:

```
  Submitted-by: david@bdt.UUCP (David Beckmeyer)
  Posting-number: Volume 1, Issue 80
  Archive-name: micro-rtx
```

The "Submitted by" is the author of the program. If you have comments about the sources published in comp.sources.atari.st this is the person to contact. When possible, this address is in domain form, otherwise it is a UUCP bang path relative to some major (backbone) site.

The second line repeats the volume/issue information for the aide of notes sites and automatic archiving programs.

The Archive-name is the "official" name of this source in the archive. Large postings will have names that look like this:

```
Archive-name: desktop/part01
```

Please try to use this name when requesting that sources be mailed to you. Also, note that the "part number" given in the title, and the archive name given in the auxiliary header need not be identical.

Since most archive sites run UNIX, articles are given UNIX-style filenames rather than ST-style filenames. I do try to limit filenames to eight characters whenever possible, though.

Subject: Reporting and tracking bugs and patches to postings

Most of the time, bugs and updates will be posted to comp.sys.atari.st.
Sometimes, "official" updates from the author will be posted submitted to
comp.sources.atari.st so that they are archived with the sources.

Since there is no easy way to apply source code patches on the ST, changes
to a program often result in a whole new version being posted.

To report bugs, contact the person listed in the Submitted-to header.
Often there is a contact address in a README file, too. I do not maintain
the sources I moderate, so don't send your bug reports to me.

Subject: Accessing the archives

At this time, there is only one site known to be keeping an archive
of comp.sources.atari.st. If you are making an archive available to
the public, or would like to, please contact me.

Subject: Listing of archive sites in no particular order

1. Site ssyx is running an archive server, which automatically answers
   mail requests. Send mail to archive-server@ssyx.ucsc.edu with the
   word "help" in the message body for more information. Both the sources
   and binaries groups are archived on ssyx.

From 8111416640508491121
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,59fd347383cdae4c,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-05-15 14:49:00 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!rutgers!tut.cis.ohio-state.edu!ucbvax!
agate!saturn!koreth@ssyx.ucsc.edu
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i039: which -- Find programs using PATH env. variable
Keywords: shar, C
Message-ID: <7485@saturn.ucsc.edu>
Date: 15 May 89 21:49:00 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 121
Approved: koreth@ssyx.ucsc.edu

Submitted-by: ravi@mcnc.org (Ravi Subrahmanyan)
Posting-number: Volume 2, Issue 39
Archive-name: which

Here's a trivial utility for the ST: basically the same as "which"
under unix.

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      README
#      WHICH.C
#
sed 's/^X//' << 'SHAR_EOF' > README &&
XThis is an ST adaptation of a "which" program somone posted to
Xnet.sources many years ago to replace the agonisingly slow
XBSD "which" script with a program.  I've used  it on our un*x
Xsystem for years, and finally decided I couldn't do without
Xit on the ST any longer..  Anyway, here it is (with source,
Xits trivial enough.)  To use it just say
X
X% which haha
X
Xand it'll add in the ".prg" or ".tos" or ".ttp" while checking for the file.
X(it'll look for all three.)  It works with the Mark Williams C "msh" PATH,
Xbut you can easily modify it for your favourite shell, just change the comma
Xin the #define for DELIMETERat the top to whatever the delimeter is for
Xthe PATH in your shell.
X
X-ravi
X(ravi@mcnc.org)
SHAR_EOF
chmod 0600 README || echo "restore of README fails"
sed 's/^X//' << 'SHAR_EOF' > WHICH.C &&
X#define DELIMETER ','
X
X#include <stdio.h>
X
Xchar *getenv();
Xchar *index();
X
Xint
Xmain(ac,av)
Xchar **av;
X{
X    char *path, *cp;
X    char buf[200];
X    char prog[200];
X    char patbuf[512];
X    int quit, none;
X
X    if (ac < 2) {
X        fprintf(stderr, "Usage: %s cmd [cmd, ..]\n", *av);
X        exit(1);
X    }
X    av[ac] = 0;
```

```
X    for(av++ ; *av; av++) {
X
X        quit = 0;
X        none = 1;
X    if ((path = getenv("PATH")) == NULL) {
X        fprintf(stderr, "Null path.\n");
X        exit(0);
X    }
X        strcpy(patbuf, path);
X        path = patbuf;
X        cp = path;
X
X        while(1) {
X            cp = index(path, DELIMETER);
X            if (cp == NULL)
X                quit++;
X            else
X                *cp = '\0';
X
X        if (strcmp(path,"") == (char *)NULL && quit == 0) {
X         sprintf(buf, "%s.\\%s", path, *av);
X        } else
X         sprintf(buf, "%s\\%s", path, *av);
X
X/* fprintf(stderr,"Trying %s, path %s\n",buf,path); */
X
X            path = ++cp;
X
X            if (access(buf, 1) == 0) {
X                printf("%s\n", buf);
X                none = 0;
X            }
X
X            sprintf(prog, "%s.%s", buf, "prg");
X            if (access(prog, 1) == 0) {
X                printf("%s\n", prog);
X                none = 0;
X            }
X
X            sprintf(prog, "%s.%s", buf, "ttp");
X            if (access(prog, 1) == 0) {
X                printf("%s\n", prog);
X                none = 0;
X            }
X
X            sprintf(prog, "%s.%s", buf, "tos");
X            if (access(prog, 1) == 0) {
X                printf("%s\n", prog);
X                none = 0;
X            }
X            if (quit) {
X                if (none)
X                    printf("No %s in %s\n", *av, getenv("PATH"));
```

```
X                break;
X            }
X        }
X    }
X    exit(0);
X}
X
X
SHAR_EOF
chmod 0600 WHICH.C || echo "restore of WHICH.C fails"
exit 0


From 6255229926254212650
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,4d8c3daa271eff80,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-05-18 23:57:05 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!rutgers!tut.cis.ohio-state.edu!ucbvax!
agate!saturn!koreth@ssyx.ucsc.edu
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i040: atime89 -- Set GEMDOS time from keyboard clock
Keywords: shar, C
Message-ID: <7562@saturn.ucsc.edu>
Date: 19 May 89 06:57:05 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 272
Approved: koreth@ssyx.ucsc.edu

Submitted-by: radoslav@kuling.docs.uu.se (Radoslav Bogdanovic)
Posting-number: Volume 2, Issue 40
Archive-name: atime89

This is the sources for the AUTO - TIMESET program which was posted to
the comp.binaries.atari.st.

---------------------- CUT HERE ----------------------------------------

: This is a shar archive.  Extract with sh, not csh.
: This archive ends with exit, so do not worry about trailing junk.
echo 'Extracting a_time89.doc'
sed 's/^X//' > a_time89.doc << '+ END-OF-FILE a_time89.doc'
X
X                    AUTO - TIMESET Version 1.0
X                 (c) by Radoslav Bogdanovic / RadoSoft
X                            17 may, 1989
X
X                For Microdeal/Michtron Internal Clock Card
X                                  and
X                Other Clock Cards working on the keyboard
X                processor for time & date maintenance.
X
```

```
X    This program works from your AUTO folder of your floppydisk or harddisk
X    and also takes over the function of the supplied AUTOTIME.PRG supplied
X    with the Microdeal/Michtron Clock Card.
X
X    The need for this program arose after I had installed the Clock Card
X    and then suddenly noticed that the clock was zeroed by some other program
X    and that I suddenly had a lot of files with wrong time and date stamp.
X    To prevent this I realised that the only way was to write a program that
X    would check if the date was somehow wrong. This is performed by checking
X    if the IKBD setting of the year is different of the xx year in the file
X    name of A_TIMExx.PRG. This mean that you only need to rename the program
X    once every year to get it to work properly.
X
X    This program was compiled with Sozobon C compiler and the complete
X    source code is supplied.
X
X    This program may freely be distributed.
X
X    I can be reached at the following address for bug reports and others:
X
X            RadoSoft
X            Radoslav Bogdanovic
X            Sernaders Vag 1-332
X            S-752 61 Uppsala
X            SWEDEN
X
X            Telephone: +46-18-463897
X
X            E-mail address: radoslav@kuling.uu.se
X
X
+ END-OF-FILE a_time89.doc
chmod 'u=rw,g=rw,o=r' 'a_time89.doc'
echo '    -rw-rw-r--  1 radoslav     1636 May 17 17:09 a_time89.doc        (as
sent)'
echo -n ' '
/bin/ls -l a_time89.doc
echo 'Extracting a_time89.c'
sed 's/^X//' > a_time89.c << '+ END-OF-FILE a_time89.c'
X/*
X    AUTO - TIMESET Version 1.0 by Radoslav Bogdanovic / RadoSoft
X            Compiled with Sozobon C -- 17 may, 1989
X
X    For Microdeal Internal Clock Card and other clock cards working
X    on the keyboard processor for time & date maintaince.
X
X    This program also takes over the function of the AUTOTIME.PRG which
X    is supplied with Microdeal Clock Card.
X
X    For more details see the supplied documentation file.
X
X*/
X
```

```
X#include <stdio.h>
X#include <ctype.h>
X#include <osbind.h>
X
Xmain(argc,argv)
Xint argc;
Xchar *argv[];
X{
X    register unsigned long IKBDtimedate;
X    register unsigned int GEMtime, GEMdate;
X    int seconds, minutes, hours, day, month, year, fyear;
X    char *ptr, ans;
X
X    printf("\n      AUTO - TIMESET Version 1.0\n");
X    printf("   by Radoslav Bogdanovic / RadoSoft\n");
X    printf("Compiled with Sozobon C -- 12 May, 1989\n");
X
X
X    /* Following works only with Sozobon C, because other C compilers does
X       NOT inserts program name (command) in ARGV[0] when program started
X       from GEM-DESKTOP or program autostarted from AUTO-folder.          */
X    ptr = argv[0];
X    while (*ptr) ++ptr;     /* Find the end of the program name */
X    ptr -= 6;               /* Back up to xx.prg                */
X    if ( isdigit(*ptr) )
X        fyear = atoi(ptr);
X    else {
X        printf("\n This is %s ...\n",argv[0]);
X        printf(" This program should be renamed to\n");
X        printf(" A_TIMExx.PRG, where xx is 80 to 99,\n");
X        printf(" two digits indicating which year\n");
X        printf(" you want to check against. \n");
X        printf("\n Press any key to exit\n");
X        Bconin(2);
X        exit(0);
X    }
X
X    IKBDtimedate = Gettime();
X    GEMtime = Tgettime();
X    GEMdate = Tgetdate();
X
X    /* Check if year from filename is same as current IKBD year  */
X    /* If year from filename differs from current IKBD year then */
X    /* it's assumed that the IKBD time is wrong or zeroed.       */
X
X    if (fyear != (((IKBDtimedate >> 25) & 0x7F) + 80)) {
X
X        IKBDtimeprint("\nThe current IKBD time and date is",IKBDtimedate);
X        timeprintGEM("The current GEM time is",GEMtime);
X        dateprintGEM("The current GEM date is",GEMdate);
X
X        printf("\nDo you want to change time and date (Y/N): ");
X        ans = Bconin(2);
```

```
X        if ((ans == 'Y')||(ans == 'y'))
X        {
X            printf("\nEnter the new date and time (YY/MM/DD HH:MM): ");
X            scanf("%d/%d/%d %d:%d",&year,&month,&day,&hours,&minutes);
X            seconds = 0;
X            if (year < 100) year += 1900;
X
X            IKBDtimedate =  ((unsigned long)(year-1980)<<25)
X                            |((unsigned long)month<<21)
X                            |((unsigned long)day<<16)
X                            |((unsigned long)hours<<11)
X                            |((unsigned long)minutes<<5)
X                            |((unsigned long)seconds<<1);
X
X            GEMdate = ((unsigned)(year - 1980)<<9)
X                     |((unsigned)month<<5)
X                     |(unsigned)day;
X
X            GEMtime = ((unsigned)hours<<11)
X                     |((unsigned)minutes<<5)
X                     |((unsigned)seconds>>1);
X
X            Settime(IKBDtimedate);
X            Tsettime(GEMtime);
X            Tsetdate(GEMdate);
X        }
X        else
X           setsystemtime(Gettime());
X        putchar('\n');
X    }
X    else
X       setsystemtime(Gettime());
X}
X
X/*-------------------------------------------------------*/
X/* fixdigit insert a '0' before number if number < 10  */
X/*-------------------------------------------------------*/
Xfixdigit(buf, number)
Xchar *buf; int number;
X{
X    if (number < 10) {
X        buf[0] = '0';
X        sprintf(&buf[1],"%d",number);
X    }
X    else
X        sprintf(buf,"%d",number);
X}
X
X/*-------------------------------------------------------------*/
X/* IKBDtimeprint decodes IKBD time & date  and prints a string   */
X/*-------------------------------------------------------------*/
XIKBDtimeprint(string,time)
Xchar *string; register unsigned long time;
```

```
X{
X    int seconds, minutes, hours, day, month, year;
X    char mins[3], secs[3], days[3], monts[3];
X
X    seconds = (time & 0x001F) << 1;
X    minutes = (time >> 5) & 0x3F;
X    hours   = (time >> 11) & 0x1F;
X
X    day     = (time >> 16) & 0x1F;
X    month   = (time >> 21) & 0x0F;
X    year    = ((time >> 25) & 0x7F) + 1980;
X
X    fixdigit(mins, minutes);
X    fixdigit(secs, seconds);
X    fixdigit(days, day);
X    fixdigit(monts, month);
X    printf("%s %d:%s:%s on %d/%s/%s\n",string,hours,mins,
X            secs, year,monts,days);
X}
X
X/*-------------------------------------------------------*/
X/* timeprintGEM decodes GEM time and prints a string   */
X/*-------------------------------------------------------*/
XtimeprintGEM(string,time)
Xchar *string; register unsigned int time;
X{
X    int seconds, minutes, hours;
X    char mins[3], secs[3];
X
X    seconds = (time & 0x001F) << 1;
X    minutes = (time >> 5) & 0x3F;
X    hours   = (time >> 11) & 0x1F;
X
X    fixdigit(mins, minutes);
X    fixdigit(secs, seconds);
X    printf("%s %d:%s:%s\n",string,hours,mins,secs);
X}
X
X/*-------------------------------------------------------*/
X/* dateprintGEM decodes GEM date and prints a string   */
X/*-------------------------------------------------------*/
XdateprintGEM(string,date)
Xchar *string; register unsigned int date;
X{
X    int day, month, year;
X    char days[3], monts[3];
X
X    day     = date & 0x1F;
X    month   = (date >> 5) & 0x0F;
X    year    = ((date >> 9) & 0x7F) + 1980;
X
X    fixdigit(days, day);
X    fixdigit(monts, month);
```

```
X    printf("%s %d/%s/%s\n",string,year,monts,days);
X}
X
X/*------------------------------------------------------------------------*/
X/* setsystemtime sets the GEM system time & date from IKBD time & date */
X/*------------------------------------------------------------------------*/
Xsetsystemtime(time)
Xregister unsigned long time;
X{
X    int seconds, minutes, hours, day, month, year;
X    unsigned int GEMtime, GEMdate;
X
X    seconds = (time & 0x001F) << 1;
X    minutes = (time >> 5) & 0x3F;
X    hours   = (time >> 11) & 0x1F;
X
X    day     = (time >> 16) & 0x1F;
X    month   = (time >> 21) & 0x0F;
X    year    = (time >> 25) & 0x7F;
X
X    GEMdate = ((unsigned)year<<9)
X              |((unsigned)month<<5)
X              |(unsigned)day;
X
X    GEMtime = ((unsigned)hours<<11)
X              |((unsigned)minutes<<5)
X              |((unsigned)seconds>>1);
X
X    Tsettime(GEMtime);
X    Tsetdate(GEMdate);
X}
X
+ END-OF-FILE a_time89.c
chmod 'u=rw,g=rw,o=r' 'a_time89.c'
echo '   -rw-rw-r--  1 radoslav     6693 May 17 17:00 a_time89.c        (as sent)'
echo -n ' '
/bin/ls -l a_time89.c
exit 0


From -5029470138514135684
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,4ae829a9657b6026,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1987-12-25 07:12:22 PST
Path: utzoo!mnetor!uunet!husc6!rutgers!ames!ucbcad!ucbvax!decvax!decwrl!sgi!daisy!
atari-sources-request
From: atari-sources-request@daisy.UUCP
Newsgroups: comp.sources.atari.st
Subject: v00i051:  mwc:all: DU - un*x like disk usage reporting program
Message-ID: <776@daisy.UUCP>
Date: 25 Dec 87 15:12:22 GMT
Sender: turner@daisy.UUCP
```

Lines: 176
Approved: daisy!turner

Submitted by: atari!imagen!ucbvax!ssyx!koreth (Steven Grimm)
comp.sources.atari.st: Volume 0, Issue 51
Archive-name: du.c

-+=+-

[Editor's Note]
     this was a simple, elegant program until i got my hacking little hands on it. Anyway it now supports the -s flag and directory names as arguements. If you don't have the man entry for du, this program reports disk usage for a directory and it's subdirectories (if any). The -s flag suppresses reporting subdirectories. The checksum is:

bytes =   3428(3584)   cksm = 666   du.c

-+=+-

Here's a submission to comp.sources.atari.st. Use it in good health.

```
#include <stdio.h>
#include <osbind.h>

/*
** DU.C
**
** Shows disk usage in a directory tree.
**
** Written by Steven Grimm, 11-19-87
**
** Usage:
**  du.ttp
*/

typedef struct {
        char    junk[21];
        char    attrib;
        int     time;
        int     date;
        long    length;
        char    name[14];
} DTA;

extern int getopt(), optind;
extern char *optarg;
int aflag, sflag;


/* Somewhat specialized long-to-ascii converter. */
char *ltoa(num,string)
```

```
register long num;
register char *string;
{
        register int ptr=0, flag=0;
        register long div=1000000000L;
        if (num<0)
        {
                string[ptr++]='-';
                num=0-num;
        }
        while ((div /= 10) >0)
        {
                if ((!(num/div))&&(!flag))
                        ++ptr;
                else
                {
                        string[ptr++]=num/div+48;
                        num %= div;
                        ++flag;
                }
        }
        if (! flag)
                string[ptr++]='0';
        string[ptr++]=' ';
        string[ptr]=0;
        return string;
}

/*
** Find disk usage in a directory.  Returns number of 1K blocks in the
** directory and its descendants.
**
** Input:
**      dir     Directory to search, ending in "\"
*/
long du(dir)
char *dir;
{
        DTA     dta;
        long    total;          /* Running total of disk usage */
        int     fstat;          /* Status returned from Fs{first,next}() */
        char    filespec[256];  /* Path and wildcard to search on */

        Fsetdta(&dta);
        strcpy(filespec, dir);
        strcat(filespec, "*.*");
        total = 0L;

        fstat = Fsfirst(filespec, 0x3f);

        while (fstat == 0)
        {
                if (dta.attrib & 0x10)  /* Search subdirectories */
```

```
                {
                        if (strcmp(dta.name, ".") && strcmp(dta.name, ".."))
                        {
                                strcpy(filespec, dir);
                                strcat(filespec, dta.name);
                                strcat(filespec, "\\");
                                total += du(filespec);
                                Fsetdta(&dta);
                        }
                }
                else
                        total += (dta.length+1023) / 1024;
                fstat = Fsnext();
        }
     if(!sflag) {
             strcpy(filespec, "                 ");
              Cconws(ltoa(total, filespec));
             Cconws(dir);
              Cconout(13);
             Cconout(10);
     }
        return total;
}

main(argc,argv)
int argc;
char *argv[];

{
     char *disk, dum[255];
     int c;
     long total;

     if(argc == 0) {
          fprintf(stderr, "Usage is: du [-a] [-s] directory\n");
          exit(1);
     }

     sflag = 0;
     aflag = 0;

     while((c = getopt(argc, argv, "as")) != EOF)
          switch(c) {
               case 'a':
                    aflag++;
                    break;
               case 's':
                    sflag++;
                    break;
               default:
                    fprintf(stderr, "Illegal option '%c'\n", c);
                    break;
          }
```

```
     if(optind >= argc) {
          strcpy(disk, ".\\");
             total = du(disk);
     }
     else {
          strcpy(disk,argv[optind]);
          if(disk[strlen(disk) - 1] != '\\')
               strcat(disk,"\\");
          total = du(disk);
     }
     if(sflag) {
             strcpy(dum, "                  ");
              Cconws(ltoa(total, dum));
             Cconws(disk);
              Cconout(13);
             Cconout(10);
     }

     exit(0);
}
```

```
From 2617288392245660340
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,bded0d17d53f95e2,start
X-Google-Attributes: gid100493,public
X-Google-Thread: 111a55,bded0d17d53f95e2,start
X-Google-Attributes: gid111a55,public
X-Google-ArrivalTime: 1988-03-20 19:35:03 PST
Xref: utzoo comp.sources.atari.st:54 comp.sys.atari.st:8458
Path: utzoo!mnetor!uunet!husc6!mit-eddie!ll-xn!ames!pasteur!ucbvax!agate!saturn!
ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st,comp.sys.atari.st
Subject: v01INF1: Introduction to comp.sources.atari.st
Message-ID: <2447@saturn.ucsc.edu>
Date: 21 Mar 88 03:35:03 GMT
Expires: 19 Jul 88 03:35:02 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 143
Approved: koreth@ssyx.UCSC.EDU

Submitted-by: koreth@ssyx.ucsc.edu (Steven Grimm)
Posting-number: Volume 1, Info 1
Archive-name: intro

This is the first of two introductory articles about comp.sources.atari.st.
This one describes how to submit source to the newsgroup, where the
archive sites are, and how to contact them.  The companion articles lists
all previously-published sources.

I am always looking for suggestions on how to improve the usefulness
of the newsgroup, and can be contacted as listed below.
```

-- Steven Grimm

Subject: Submitting source for publication

Items intended for posting or queries and problem notes should be sent to atari-sources@ssyx.ucsc.edu. If you are on a UUCP-only site, you can send them to ucbvax!ssyx!atari-sources.

If you want verification of arrival, so say in a cover note, or at the beginning of your submission, if it is small. I try to verify that a program works, and if I can't get it to work, I may hold up posting it for a couple of days. Please note that, except in rare cases, source without documentation will not be published. The backlog from receipt to posting varies from one to four weeks depending mostly on the set of submissions currently in my queue.

Please inform me of any special system requirements, and what development environment was used, so that I can properly test your submission.

Subject: The structure of comp.sources.atari.st articles

Each posting in comp.sources.atari.st is called an "issue"; there are roughly 100 issues to a volume. The division is arbitrary and may vary. There are two types of articles in comp.sources.atari.st: sources and "information postings." They can be distinguished by the subject line:

  Subject: v01INF3: Index for Volume 1 and other info

This first word in the title identifies this as the third info posting of volume one. Similarly, the subject line shown below:

  Subject: v01i081: Public Domain Resource Construction Set

identifies this as the 81st source article in Volume 1. Large sources are broken up into smaller pieces, and have subject lines that look like this:

  Subject: v01i082: VMS clone in ST-BASIC, Part03/08

Certain information about the system configuration required to use the program is given on the keywords line.

  Keywords: MWC, Megamax, 1meg, high

This means that the program compiles with Mark Williams or Megamax C, requires at least one meg of RAM and only runs in high resolution. Following is a list of keywords; new ones may be added as needed. They are mostly self-explanatory.

```
  Alcyon  - Alcyon C (included with Atari developer's kit)
  MWC         - Mark Williams C
  Megamax - Megamax C
  Laser       - Laser C
  Lattice - Lattice C
  OSI         - OSI Personal Pascal
  BASIC       - ST BASIC (this is not a joke!)
  uuencode    - program is uuencoded (uudecode required to unpack)
  arc         - program is archived (arc required to unpack)
  high        - high resolution
  medium  - medium resolution
  low         - low resolution
  1meg        - needs 1 meg of RAM
```

The first few lines of an article are auxiliary headers that look like this:

```
  Submitted-by: david@bdt.UUCP (David Beckmeyer)
  Posting-number: Volume 1, Issue 80
  Archive-name: micro-rtx
```

The "Submitted by" is the author of the program. If you have comments about the sources published in comp.sources.atari.st this is the person to contact. When possible, this address is in domain form, otherwise it is a UUCP bang path relative to some major (backbone) site.

The second line repeats the volume/issue information for the aide of notes sites and automatic archiving programs.

The Archive-name is the "official" name of this source in the archive. Large postings will have names that look like this:

```
Archive-name: desktop/part01
```

Please try to use this name when requesting that sources be mailed to you. Also, note that the "part number" given in the title, and the archive name given in the auxiliary header need not be identical.

Since most archive sites run UNIX, articles are given UNIX-style filenames rather than ST-style filenames.

Subject: Reporting and tracking bugs and patches to postings

Most of the time, bugs and updates will be posted to comp.sys.atari.st. Sometimes, "official" updates from the author will be posted submitted to comp.sources.atari.st so that they are archived with the sources.

Since there is no easy way to apply source code patches on the ST, changes to a program often result in a whole new version being posted.

To report bugs, contact the person listed in the Submitted-to header.

Often there is a contact address in a README file, too. I do not maintain
the sources I moderate, so don't send your bug reports to me.

Subject: Accessing the archives

At this time, there is only one site known to be keeping an archive
of comp.sources.atari.st. If you are making an archive available to
the public, or would like to, please contact me.

Subject: Listing of archive sites in no particular order

1. Site killer has a full archive available for anonymous UUCP. For
   information on accessing it, send mail to Bill Wisner:
   {ames,codas,ihnp4,mit-eddie}!killer!wisner.

+New! Improved! Now 200% Artificial-+-+-----------Steven Grimm------------+
|#   # @@@ **** &&&&& $$$$$ %   %| | ARPA: koreth@ssyx.ucsc.edu |
|# # @   @ *   * & $ %   %+-+ UUCP: ...!ucbvax!ucscc!ssyx!koreth|
|###   @   @ **** &&&& $ %%%%%| |_____________________________________|
|# # @   @ * *   & $ %   %+-+"Things are only impossible until |
|#   # @@@ * ** &&&&& $ %   %| | they're not." - Jean-Luc Picard |
+-----with NutraSour(TM)! No natural colors or preservatives!------------+

From 1916076722437212858
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,939a5ceba12a8e10,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-03-20 19:49:14 PST
Path: utzoo!mnetor!uunet!husc6!mit-eddie!ll-xn!ames!pasteur!ucbvax!agate!saturn!
ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i001: uudecode.bas - starter version of uudecode in BASIC
Message-ID: <2449@saturn.ucsc.edu>
Date: 21 Mar 88 03:49:14 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 108
Keywords: BASIC
Approved: koreth@ssyx.ucsc.edu

Submitted-by: wisner@eddie.MIT.EDU (Bill Wisner)
Posting-number: Volume 1, Issue 1
Archive-name: basic-uud

[This is a uudecode program written in ST BASIC and is revoltingly slow.
It should be used to decode the better, faster, compiled version called uud
that will follow in comp.binaries.atari.st. This version will only work
with files uuencoded by UNIX uuencode. ..b]

```
10    ' UUDECODE PROGRAM
11    ' /*
12    '  * Uudecode -- decode a uuencoded file back to binary form
13    '  *
14    '  * Slightly modified from a version posted to net.sources once;
15    '  * suitable for compilation on an IBM PC.
16    '  *
17    '  * modified for Lattice C on the ST - 11.05.86 by MSD
18    '  *
19    '  * rewritten for ST BASIC on the Atari ST - September 9, 1986,
20    '  *   by Patrick Simon
21    '  *
22    '  */

23    ' Initializations.
24    DIM I(4)
25    FULLW 2 : CLEARW 2 ' Use full output window, clear it.
26    MASK = 63 : SP = ASC(" ")

30    ' Get input filename and open file.
40    INPUT "Enter name of uuencoded file > ", IFILEN$
50    OPEN "I", #1, IFILEN$

60    ' Get output filename, open, and set up pointers.
70    INPUT "Enter name of output file > ", OFILEN$
80    OPEN "R", #2, OFILEN$, 45
90    FIELD #2, 45 AS CBUF$

110   ' Input and ignore first line of input file (original filename).
120   LINE INPUT #1, INLINE$ ' Due to a bug in ST BASIC, LINE INPUT #
130   LINE INPUT #1, INLINE$ '  must be called 2x, just to get first
140                          '  line. Even then we lose the 1st character.

150   ' Now set up screen for user.
160   CLEARW 2
170   GOTOXY 1,2 : PRINT "Decoding "; IFILEN$; " to "; OFILEN$
180   GOTOXY 1,4 : PRINT "# lines read in: "
190   GOTOXY 1,5 : PRINT "# bytes output: "

200   ' Initialize some variables for main body of program.
210   NUML = 0 : BYTEOUT = 0

211   ' [Comments removed from middle of loop to increase execution speed]
212   ' LINE INPUT should work OK by now!
213   ' (l. 231)  check to see if we are done yet
214   ' (general) For each line, must "decode" each byte.
215   '           First byte contains # bytes to ***output*** from this line.
216   '           Then, after decoding, each set of 4 input bytes contains 6*4 =
 24
217   '           bits of information to output, 8 bits per output byte, thus
218   '           3 output bytes.
219   ' (l. 260)  SPTR is string position pointer for INLINE$
```

```
220   ' (l. 280)  Get first byte = # bytes to output;
221   '           exit if line has zero to output
222   ' (l. 290)  Process one line of input
223   ' (l. 310)  Decode set of 4 input bytes.
224   ' (l. 410)  If more in this line, go back again.
225   ' (l. 420)  Finished processing line; ouput it; show status to user
226   ' (l. 450)  Get next line ...

228   LINELOOP: ' Now loop for as long as there are input lines.
230      LINE INPUT #1, INLINE$
231      IF INLINE$ = "end" THEN GOTO 500
232      NUML = NUML + 1

260      SPTR = 1

280      GOSUB DECODE : NN = C : N = NN : IF NN = 0 THEN GOTO 500

290      MPTR = 1
300      WHILE N > 0
310         FOR J = 1 TO 4
320            GOSUB DECODE : I(J) = C
330         NEXT J
340         C1 = ((I(1) *  4) MOD 256) + INT(I(2) / 16)
350         C2 = ((I(2) * 16) MOD 256) + INT(I(3) /  4)
360         C3 = ((I(3) * 64) MOD 256) + I(4)
370         IF N>0 THEN MID$(CBUF$,MPTR,1)=CHR$(C1) : MPTR = MPTR + 1
380         IF N>1 THEN MID$(CBUF$,MPTR,1)=CHR$(C2) : MPTR = MPTR + 1
390         IF N>2 THEN MID$(CBUF$,MPTR,1)=CHR$(C3) : MPTR = MPTR + 1
400         N = N - 3
410      WEND

420      PUT #2, NUML
430      BYTEOUT = BYTEOUT + NN
440      GOTOXY 20,4 : PRINT NUML
445      GOTOXY 20,5 : PRINT BYTEOUT
450   GOTO LINELOOP

500   ' Hopefully, we have now decoded the input file.
510   CLOSE #1 : CLOSE #2
520   END

999   ' Subroutine to decode a single character
1000  DECODE:
1010     IF LEN(MID$(INLINE$,SPTR)) = 0 THEN INLINE$ = INLINE$ + " "
1020     CH$ = MID$(INLINE$, SPTR, 1) : SPTR = SPTR + 1
1030     C = (ASC(CH$) - SP) AND MASK
1040  RETURN
```


From 2681857747314377300
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,3ab301e14b6d5c08,start
X-Google-Attributes: gid100493,public

X-Google-ArrivalTime: 1988-03-28 10:41:28 PST
Path: utzoo!mnetor!uunet!husc6!mailrus!ames!pasteur!ucbvax!agate!saturn!ssyx!
koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i002: Automatic file copier for RAMdisk
Message-ID: <2531@saturn.ucsc.edu>
Date: 28 Mar 88 18:41:28 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 361
Keywords: Laser, Megamax
Approved: koreth@ssyx.ucsc.edu

Submitted-by: sirkm@ucscb.ucsc.edu (Greg Anderson)
Posting-number: Volume 1, Issue 2
Archive-name: bootcopy

[This is rather timely, as discussions about just this sort of program have
 been going on in comp.sys.atari.st. It should work with any C compiler,
 but has only been tested with Laser and Megamax C. -sg]

```
/*--------------------------------------------------------------------
| Bootcopy version 1.0:   In the Public Domain
|        Greg Anderson
|
| This program copies everything inside the folder "AUTOCOPY" onto
| the Ramdisk on powerup.  If "AUTOCOPY" contains folders, these
| folders are created on the Ramdisk, and their contents are also
| copied.  There is no limit to the depth of folders that can be
| placed inside "AUTOCOPY".
|
| Usage:
|        Putting Bootcopy into the AUTO folder:
|
| 1. If you already have an AUTO folder on your disk, save the files
|    inside it somewhere and delete it.
| 2. Create (or recreate) your AUTO folder.
| 3. Real-time-clock setting programs (such as CLKSET for DeskCart)
|    may be copied to the AUTO folder if desired.
| 4. Copy your RamDisk to the Auto folder.  Desk accessory RamDisks
|    are not suitable, as the RamDisk MUST be installed before
|    Bootcopy runs.  VEGEMATIC (MEGEMATIC.PRG) is the recommended
|    RamDisk.  Make sure that your RamDisk is the drive with the
|    highest letter, or Bootcopy will copy its files to the wrong
|    drive.
| 5. Copy BOOTCOPY.PRG to the Auto folder
| 6. Copy any other programs you wish to place in the Auto folder.
|
|        Setting up the AUTOCOPY folder:
|
| 1. Create a folder called AUTOCOPY.  It may be placed in one of
|    two locations:  on the root, or inside the AUTO folder.
| 2. Copy any files you wish to be moved to the RamDisk on powerup
```

```
|    to the AUTOCOPY folder.
|
| That's all there is to it.  The author can be reached as follows:
|
| 1. Via E-mail to sirkm@ucscb.ucsc.edu
| 2. Call his BBS (1200/300) at (408) 462-3832.
-----------------------------------------------------------------*/
#include <OSBIND.H>

int fstatus,copi,working,RamDisk;
char *bufadr,cfile[50],folder[128],*rfolder,copies[30],str[30];
long buflen;

main()
{
     print("~---------------------------------~");
     print("       Bootcopy vrs. 1.0,~");
     print("    a Public Domain Utility~");
     print("        by Greg Anderson~");
     print("---------------------------------~");

     /*-------------------------------------------------------------
     | Assume RamDisk is the highest-lettered drive installed
     -------------------------------------------------------------*/
     fstatus=-1;
     for(RamDisk=16;fstatus<0;--RamDisk)
     {
          Dsetdrv(RamDisk-1);
          fstatus=Dsetpath("\\");
     }
     if(RamDisk<2)
          print("~~No RamDisk handler installed.~~");
     else
     {
          strcpy(str,"~RamDisk = D:~");
          str[11]=RamDisk+65;
          print(str);
     }
     strcpy(copies,"   0 Files copied.");
     copi=3;
     /*-------------------------------------------------------------
     | Find free memory & allocate it for the copy buffer
     -------------------------------------------------------------*/
     buflen=Malloc(-1L);
     bufadr=(char *)Malloc(buflen-2048);
     /*-------------------------------------------------------------
     | Start inside the folder "\AUTOCOPY"
     -------------------------------------------------------------*/
     strcpy(folder,"\\AUTOCOPY\\");
     rfolder = folder+9;
     working=1;
     /*-------------------------------------------------------------
     | Does "\AUTOCOPY exist?
```

```
     ------------------------------------------------------------*/
     Dsetdrv(0);
     fstatus=Dsetpath(folder);
     if(fstatus)
     {
          /*-------------------------------------------------
          | It is also permissible to place AUTOCOPY inside of
          |     the AUTO folder to avoid cluttering the root.
          -------------------------------------------------*/
          strcpy(folder,"\\AUTO\\AUTOCOPY\\");
          rfolder = folder+14;
          fstatus=Dsetpath(folder);
     }
     /*-------------------------------------------------------------
     | Exit if there is no AUTOCOPY folder.
     -------------------------------------------------------------*/
     if(fstatus)
     {
          print("~~No AUTOCOPY folder found on this disk.~");
     }
     if(fstatus | (RamDisk<2))
     {
          print("Create an AUTOCOPY folder on drive A:\\~");
          print("and place this program inside the AUTO~");
          print("folder after the RamDisk handler.  All~");
          print("files in AUTOCOPY will then be moved to~");
          print("the RamDisk on powerup.~~");
          print("Press a key:~");
          Cconin();
     }
     else
     {
          while (working)
          {
               filecopy();
               nextfolder();
          }
          print("~~Bootcopy complete. ");
          print(copies);
     }
     /*-------------------------------------------------------------
     | Return copy buffer to the system
     -------------------------------------------------------------*/
     buflen=Mfree(bufadr);
}

/*-------------------------------------------------------------------
| Copy all files in the subdirectory "folder" to the Ramdisk
-------------------------------------------------------------------*/
filecopy()
{
     int rhand,whand,i;
     long len;
```

```
char name[128];

/*-----------------------------------------------------------
| Set the working directory on the RamDisk to "rfolder"
|     & make the working directory on drive A "folder".
|     Note that "rfolder" points to the same memory space
|     as "folder" (but past the "\AUTOCOPY" header).
-----------------------------------------------------------*/
Dsetdrv(RamDisk);
Dsetpath(rfolder);
Dsetdrv(0);
Dsetpath(folder);
fstatus=Fsfirst("A:*.*",0);

while (! fstatus)
{
     /*-------------------------------------------------
     | Find the next file in the directory & open for read
     -------------------------------------------------*/
     filename(cfile);
     strcpy(name,"A:");
     strcat(name,cfile);
     rhand=Fopen(name,0);
     if(rhand>-1)
     {
          /*----------------------------------------
          | Create the file on the RamDisk
          ----------------------------------------*/
          name[0] = 65 + RamDisk;
          whand=Fcreate(name,0);
          if(whand>-1)
          {
               /*-----------------------------------
               | Copy the file from rhand to whand
               -----------------------------------*/
               do
               {
                    len=Fread(rhand,buflen,bufadr);
                    Fwrite(whand,len,bufadr);
               } while(len==buflen);
          }
     }
     /*-------------------------------------------------
     | Copy complete--increment the copied count
     -------------------------------------------------*/
     ++copies[copi];
     i=copi;
     while(copies[i]==58)
     {
          copies[i--]=48;
          if(copies[i]<48)
               copies[i]=48;
          ++copies[i];
```

```
          }
          fstatus=Fsnext();
     }
}

/*-----------------------------------------------------------------
| Determine if there are folders in this folder that have not been
| coppied.  If there are, set the working directory, create a like
| directory on the RamDisk and return.  Otherwise, try to go up a     *
| level.  If not possible, then set 'working' to zero to indicate
| failure.  If it works, call this routine again.
-----------------------------------------------------------------*/
nextfolder()
{
     int loop,i,check;
     char savename[16],cmpname[16],newfolder[16],fspec[128];
     loop=1;
     while (loop)
     {
          /*-------------------------------------------------
          | Read past all of the folders that have already
          |     been copied onto the Ramdisk
          -------------------------------------------------*/
          Dsetdrv(0);
          Dsetpath(folder);
          strcpy(fspec,"A:");
          strcat(fspec,folder);
          strcat(fspec,"*.*");
          fstatus=Fsfirstd(fspec);
          check=0;

          while((fstatus == 0) & (check==0))
          {
               filename(cmpname);
               strcpy(fspec,rfolder);
               strcat(fspec,cmpname);
               strcat(fspec,"\\");
               Dsetdrv(RamDisk);
               check=Dsetpath(fspec);
               if(check>-1)
                    fstatus=Fsnextd();
          }
          /*-------------------------------------------------
          | Reset working directory to where it was
          -------------------------------------------------*/
          Dsetdrv(RamDisk);
          Dsetpath(rfolder);
          /*-------------------------------------------------
          | Are there any more folders?  If so, create the
          |     next one & append its name to "folder" so it
          |     will be copied next time through the loop
          -------------------------------------------------*/
          if (fstatus==0)
```

```
          {
               Dsetdrv(RamDisk);
               Dcreate(cmpname);
               strcat(folder,cmpname);
               strcat(folder,"\\");
               loop=0;
          }
          else
          {
               /*----------------------------------------
               | If there are no more folders to copy on
               |    this level & if we are on the root, then
               |    the copy is finished.
               ----------------------------------------*/
               i = strlen(rfolder)-2;
               if(i < 0)
               {
                    working = 0;
                    loop = 0;
               }
               /*----------------------------------------
               | If there are no more folders to copy on
               |    this level but we are not on the root,
               |    then go up one level and see if there
               |    are any folders there that need copying
               ----------------------------------------*/
               else
               {
                    while(rfolder[i] != 92)
                         --i;
                    rfolder[i+1]=0;
               }
          }
     }
}

/*---------------------------------------------------------------------
| Extract the filename from the DTA
---------------------------------------------------------------------*/
filename(string)
char *string;
{
     char *ptr;
     ptr=(char *)(Fgetdta()+30);
     strcpy(string,ptr);
}

/*---------------------------------------------------------------------
| These two routines work like Fsfirst and Fsnext, but they skip
|     any entries that are not folders.
---------------------------------------------------------------------*/
Fsnextd()
{
```

```
     char *ptr;
     int stat;
     do
     {
          stat=Fsnext();
          ptr=(char *)(Fgetdta()+21);
     } while((isfolder() == 0) & stat == 0);
     return(stat);
}

Fsfirstd(fspec)
char *fspec;
{
     int stat;
     stat=Fsfirst(fspec,16);
     if(isfolder() == 0)
          stat=Fsnextd();
     return(stat);
}

/*----------------------------------------------------------------------
| Return true if the file is a folder.
----------------------------------------------------------------------*/
isfolder()
{
     char *ptr;
     ptr=(char *)(Fgetdta());
     if(ptr[30]==46)                     /* Ignore folders */
          return(0);                  /* that start with "." */

     return((ptr[21] && 16) != 0);
}

/*----------------------------------------------------------------------
| A simple print routine.  Convert ~ to CR LF.  This routine is
|     used to prevent the large printf() library from linking.
----------------------------------------------------------------------*/
print(string)
char *string;
{
     char c;
     int i;

     i=0;

     while(c=string[i++])
     {
          if(c==126)
          {
               Cconout(13);
               Cconout(10);
          }
          else
```

```
                Cconout(c);
        }
}
```

```
From 6717555125481586601
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,d3ee4204764f759a,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-03-29 09:58:37 PST
Path: utzoo!mnetor!uunet!husc6!think!ames!pasteur!ucbvax!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i003: Joystick-reading routines
Message-ID: <2544@saturn.ucsc.edu>
Date: 29 Mar 88 17:58:37 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 198
Keywords: Laser
Approved: koreth@ssyx.ucsc.edu

Submitted-by: sirkm@ucscb.ucsc.edu (Greg Anderson)
Posting-number: Volume 1, Issue
Archive-name: joyread

[Lots of STers seem to have trouble reading the joystick ports, which is much
 more painful on the ST than on the 8-bit.  These routines should help make
 it easier.  Note that these are just some routines, and not a complete
 program. -sg]

/*-----------------------------------------------------------------------
| Stick.c:  Joystick interupt service routine
|
| Placed in the public domain March 1988 by Greg Anderson.
|
|      The joystick demo by Steve Jankowski was instrumental to
|                  the creation of these routines.
|
|   Many thanks to Steven Grimm for providing essential information
|               and the aforementioned joystick demo.
|
|#######################################################################
| THESE ROUTINES ONLY WORK UNDER LASER C!  THEY MUST BE MODIFIED FOR
| OTHER C COMPILERS, _INCLUDING_ MEGAMAX C VERSION 1.xx!
|
| MegaMax instructions:  A4 must be saved somewhere so that it is
|     available durring the isr.  Then, the isr can access joyinf
|     with "lea joyinf(A4),A1" instead of "lea joyinf,A1".
|#######################################################################
|
|     This package sets up an interupt to read the joystick on the
| ST.  The joystick is read via the ikbd--the intelligent keyboard
| interface chip.  The ikbd can be placed into "joystick-reading
```

```
| mode" by sending an appropriate command with Ikbdws().  This has
| two effects:  first, the mouse is disabled.  Second, the ikbd will
| be prompted to send a three-byte packet every time one of the
| joysticks changes states.  The packet is as follows:
|
|     [Stick #]  [State of Stick 0]  [State of Stick 1]
|
|     The first byte is $FE if stick 0 changed, or $FF if stick 1
| changed.  These routines ignore the first byte.
|     The next two bytes contain the states of sticks 0 and 1.
| There values will be the sum of:
|
|                  +1    (+128)
|                   |
|            +4 ---< >--- +8
|                   |
|                  +2
|
|     Add the apropriate value if the contact is down.  For example,
| if stick 0 is NW and the trigger has not been pressed, its
| current state will be 1 + 4 = 5.
|
|     These packets are sent to a routine pointed to by "joyvec".
| Joyvec is part of the structure returned by Kbdvbase().  Joyvec
| can be changed to point to an interupt service routine (isr) of
| our devising.  If this is done, the isr will find that A0 points
| to the first byte of the the packet sent by the ikbd.  The isr
| must be very short (less than 1ms exicution time).  Atari's docs
| of Kbdvbase indicate that a copy of the pointer to the packet is
| also deposited on the stack; however, this does not appear to be
| the case.
|
| Using these routines:
|
| Call stick_init() to set up the joystick isr and place the ikbd
|    into joystick send mode.  This will also disable the mouse.
|
| stick_exit() will return the ikbd to mouse send mode.  Your
|    program should do this before exiting.
|
|    Once the initialize routine has been called, the following
| commands are available:
|
| _stick(n):  Return the state of stick N.  (See above chart)
| stick(n) :  Return the direction of stick N.  (Minus the trigger)
| trig(n)  :  Return the state of the trigger on stick N.
| hstick(n):  Return the horizontal position of stick N.  (-1 to +1)
| vstick(n):  Return the vertical position of stick N.
|
-------------------------------------------------------------------------*/
#include <osbind.h>

char ikbdcmd[10];           /* Storage space for ikbd commands  */
```

```
#define SENDJOY   0x14      /* Send joystick packets           */
#define SENDMOUSE 0x08      /* Send mouse packets              */

extern joyisr();            /* Interupt service routine follows */
int (*oldjoyvec)();         /* Save old isr vector here        */

int initialized = 0;        /* Initialized flag                */
unsigned char joyinf[2];    /* Holds data for joystick states  */

int joytab[4] =    { 0, -1, 1, 0 };
#define xstick(v)  joytab[ v & 0x03 ]

/*--------------------------------------------------------------------
| Initialize joystick isr
--------------------------------------------------------------------*/
void stick_init()
{
     kbdvecs *KB;

     /* Do nothing if joystick isr has been initialized */
     if (initialized)
          return;
     /* Put our isr into the joystick interupt vector slot */
     KB = Kbdvbase();
     oldjoyvec = KB->joyvec;
     KB->joyvec = joyisr;
     /* Tell ikbd to send joystick packets. */
     ikbdcmd[0] = SENDJOY;
     Ikbdws(0, ikbdcmd);
     initialized = 1;
}

/*--------------------------------------------------------------------
| End joystick mode--go back to sending mouse packets.
--------------------------------------------------------------------*/
void stick_exit()
{
     kbdvecs *KB;

     /* Do nothing if joystick isr not initialized */
     if (! initialized)
          return;
     /* Tell ikbd to go back to sending mouse packets */
     ikbdcmd[0] = SENDMOUSE;
     Ikbdws(0, ikbdcmd);
     /* Restore the old joystick interupt vector */
     KB = Kbdvbase();
     KB->joyvec = oldjoyvec;
     initialized = 0;
}

/*--------------------------------------------------------------------
| Here is the actual joystick interupt service routine.
```

```
|    Both stick 0 and stick 1 are placed into the variable
|    "joyinf".
------------------------------------------------------------------*/
asm
{
joyisr:
     lea  joyinf,A1     /* "joyinf" holds data        */
     addq.l   #1,A0          /* We don't want first byte   */
     move.b   (A0)+,(A1)+    /* Joystick 0 state           */
     move.b   (A0)+,(A1)+    /* Joystick 1 state           */
     rts
}

/*------------------------------------------------------------------
| Return the state of stick 0 or stick 1.
|    Bits 0-3 = stick direction, bit 7 = trigger.
------------------------------------------------------------------*/
int _stick(n)
int n;
{
     return(joyinf[n]);
}

/*------------------------------------------------------------------
| Return the direction bits of stick 0 or stick 1.
------------------------------------------------------------------*/
int stick(n)
int n;
{
     return( joyinf[n] & 0x0F );
}

/*------------------------------------------------------------------
| Return the state of the trigger of stick n:
------------------------------------------------------------------*/
int trig(n)
int n;
{
     return( (joyinf[n] & 0x80) != 0 );
}

/*------------------------------------------------------------------
| Return the horizontal state of stick n:
|    +1 = right, 0 = centered, & -1 = left
------------------------------------------------------------------*/
int hstick(n)
int n;
{
     return(xstick(joyinf[n] / 4));
}

/*------------------------------------------------------------------
| Return the vertical state of stick n:
```

```
|    +1 = down, 0 = centered, -1 = up.
-----------------------------------------------------------------*/
int vstick(n)
int n;
{
     return(xstick(joyinf[n]));
}
```

```
From -276642943392891455
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,2a44732ef3444b17,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-04-18 09:31:14 PST
Path: utzoo!mnetor!uunet!husc6!cmcl2!nrl-cmf!ames!pasteur!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i016: Fast disk eraser
Message-ID: <2882@saturn.ucsc.edu>
Date: 18 Apr 88 16:31:14 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 141
Keywords: Laser
Approved: koreth@ssyx.ucsc.edu

Submitted-by: braner@tcgould.tn.cornell.edu (Moshe Braner)
Posting-number: Volume 1, Issue 16
Archive-name: blank


Here is a very simple little utility.
If you want to recycle a previously-used diskette, you can either
delete all the files and folders on it (slow) or reformat it (also
slow).  Or you can use this program, BLANK.TOS.  It zeros out the
FATs (excluding the first 3 bytes) and the root directory.  The
disk then looks just like a freshly formatted one to TOS.
You specify the drive letter, and BLANK will handle TOS or MS-DOS
disks of all sizes and formats.

The source (Laser C) follows.  It should compile with any ST C
compiler, just make sure 'WORD' is defined to 'int' or 'short', whatever
is 16 bits long.

- Moshe Braner

~~~~~~~~~~~~~~~ cut here ~~~~~~~~~~~~~~~~
/*
 *   BLANK
 *
 *   A program to blank out a floppy disk.
 *   The disk must have been formatted before.
 *   Much faster than reformatting or even
 *   deleting all that's on the disk by files.
```

```
 *
 *   For the Atari ST.
 *
 *   By Moshe Braner, 880406
 */

#include <osbind.h>
#undef  Getbpb
#define Getbpb(a)  bios(7, a)

#define WORD int   /* 16 bits, 'int' in Laser C */

#define OK    0

#define READ  0
#define WRITE 1

typedef struct bpb {
     WORD recsiz;
     WORD clsiz;
     WORD clsizb;
     WORD rdlen;
     WORD fsiz;
     WORD fatrec;
     WORD datrec;
     WORD numcl;
     WORD bflags;
} BPB;

char buf[7*512];

/*
 * Print message and quit.
 */
error(msg)
     char msg[];
{
     Cconws(msg);
     Cconws("\007\r\n\n\tHit any key ");
     Bconin(2);
     exit(0);
}

main()
{
     int  i, bps, spc, fats, dir, spd, spf, dat, sec, secs, drv;
     BPB  *bpb;

     char *msg1 = "\r\n\tError reading!";
     char *msg2 = "\r\n\tError writing!";

     Cconws("\033E\r\n\n\tBLANK 1.0\tby Moshe Braner");
```

```
loop:    Cconws("\r\n\n\tEnter letter label of drive to blank: ");
     drv = Bconin(2);
     if (drv == 0x1B)
          exit(0);
     if (drv>='a' && drv<='p')
          drv += 'A'-'a';
     Bconout(2, drv);
     if (drv < 'A' || drv > 'P')
          goto loop;
     drv -= 'A';        /* 0=A, 1=B, ... */

     /* get disk parameters from bios parameter block */

     Cconws("\r\n\tInsert disk to blank, hit any key: ");
     i = Bconin(2);
     if (i == 0x1B)
          exit(0);
     bpb = (BPB *) Getbpb(drv);
     if (bpb==0)
          error(msg1);
     dir = bpb->rdlen;      /* length of dir in sectors */
     spf = bpb->fsiz;       /* sectors per FAT     */
     dat = bpb->datrec;     /* no. of first data sector */

     /* read FAT from disk to RAM */

     sec = dat - dir - 2*spf;
     if (Rwabs (READ, buf, spf, sec, drv) != OK)
          error(msg1);

     /* zero out everything beyond the first 3 bytes */

     for (i=3; i<7*512; i++)
          buf[i] = 0;

     /* write it back, to both FATs */

     Cconws("\007\r\n\n\t\tWARNING:");
     Cconws("\r\n\tAbout to erase ALL data in drive ");
     Bconout(2,drv+'A');
     Cconws(".\r\n\n\t\tProceed (y/n)? ");
     i = Bconin(2);
     if (i!='y' && i!='Y')
          exit(0);

     if (Rwabs (WRITE, buf, spf, sec, drv) != OK)
          error(msg2);
     sec += spf;
     if (Rwabs (WRITE, buf, spf, sec, drv) != OK)
          error(msg2);

     /* zero out root directory */
```

```
     buf[0] = buf[1] = buf[2] = 0;
     sec = dat - dir;
     if (Rwabs (WRITE, buf, dir, sec, drv) != OK)
          error(msg2);

     error(".\r\n\n\tBLANK finished, no errors.");
}


From 6743716685757635759
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,ae187aaa113385f8,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-04-24 10:45:16 PST
Path: utzoo!mnetor!uunet!lll-winken!lll-lcc!ames!pasteur!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i017: Dummy routines for shrinking Megamax executables
Message-ID: <2987@saturn.ucsc.edu>
Date: 24 Apr 88 17:45:16 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 33
Keywords: Megamax
Approved: koreth@ssyx.ucsc.edu

Submitted-by: braner@tcgould.tn.cornell.edu (Moshe Braner)
Posting-number: Volume 1, Issue 17
Archive-name: mmdummy

/*
 *   MMDUMMY
 *
 *  This file defines as stubs the functions Megamax (1.1) insists on
 *  linking to all programs.  If your program does not use UNIX-like
 *  I/O, but rather calls GEMDOS, BIOS, XBIOS routines, then linking
 *  to this file (compiled) will cut the size of the executable file.
 *  For example, the program "main(){Cconws("Hello, world!\r\n");}"
 *  compiles into 410 bytes instead of 1810.  First type "mmcc mmdummy.c".
 *  Save mmdummy.o.  Later type: "mmlink myprog.o mmdummy.o".
 *
 *   - by Moshe Braner, 871212
 */

#include <osbind.h>

open(){}
lseek(){}
fgetc(){}
ungetc(){}
getw(){}
fputc(){}
putw(){}
write(){}
```

```
fflush(){}
fclose(){}
close(){}

_exit(n) int n; { Pterm(n); }
exit(n)  int n; { _exit(n); }


From 2214451367674070717
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,ceeb6bc4b03862e6,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-04-24 10:46:59 PST
Path: utzoo!mnetor!uunet!lll-winken!lll-lcc!ames!pasteur!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i018: Dummy floating-point math routines for Megamax
Message-ID: <2988@saturn.ucsc.edu>
Date: 24 Apr 88 17:46:59 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 43
Keywords: Megamax
Approved: koreth@ssyx.ucsc.edu

Submitted-by: braner@tcgould.tn.cornell.edu (Moshe Braner)
Posting-number: Volume 1, Issue 18
Archive-name: fdum

/*  A dummy fmath.c (to make fmath.o) that will satisfy the linker
    but not introduce any FP code into non-FP programs.

     By Moshe Braner, 870629.
 */

int _maxprec;
long jmp_tbl[5];
double _fpreg0, _fpreg1, _fpreg2;

_fcom(){}
_fadd(){}
_fsub(){}
_fmul(){}
_fdiv(){}
_incdouble(){}
_decdouble(){}
_incfloat(){}
_decfloat(){}
_0from_lon(){}
_0to_long(){}
_0from_flo(){}
_0to_float(){}
_1from_lon(){}
_1to_long(){}
```

```
_1from_flo(){}
_1to_float(){}
_2from_lon(){}
_2to_long(){}
_2from_flo(){}
_2to_float(){}
_ffunc(){}
_to_float(){}
_from_floa(){}
_from_long(){}
_to_long(){}
power10(){}
fconvert(){}
exponent(){}
putexp(){}


From 7909117077073779314
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,4c8b8bceec22184d,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-12 14:35:40 PST
Path: utzoo!attcan!uunet!seismo!sundc!pitstop!sun!decwrl!ucbvax!agate!saturn!ssyx!
koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i032: Disk cache system in C
Keywords: MWC
Message-ID: <3281@saturn.ucsc.edu>
Date: 12 May 88 21:35:40 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 585
Approved: koreth@ssyx.ucsc.edu

Submitted-by: egisin@watmath.waterloo.edu (Eric Gisin)
Posting-number: Volume 1, Issue 32
Archive-name: cache

[This is a cache system, which speeds up floppy accesses.  The executables
 are being posted to comp.binaries.atari.st along with the same readme file
 included here. -sg]

#--------------------------Cut Here--------------------------
#! /bin/sh
# This is a shell archive.  Remove anything before the "#! /bin/sh" line,
# then unpack it by saving it in a file and typing "sh file."
#
# unpacks with default permissions
#
# Contents : cache.c cstat.c cache.h readme
#
if `test ! -s cache.c`
then
```

```
echo "x - cache.c"
cat > cache.c << '@\Rogue\Monster\'
/*
 * Floppy Disk Cache (Atari ST)
 *
 * Copyright 1988, Eric Gisin <egisin@UWaterloo.CA>
 * this program may be copied as long as this copyright notice is retained.
 * the author will not be liable for any damages resulting from the use of this
program.
 *
 * Usage: cache [drive][size]
 *   can be loaded twice to handle two drives.
 *
 * The cache is based on paging and page replacement techniques
 * used in virtual memory systems. The replacement algorithm is
 * similiar to "not frequently used" (NFU).
 *
 * each sector of the disk has a (struct) Cache element
 * that contains a pointer to a cache buffer,
 * and an "average usage" value.  The average is a combination
 * of long term usage and short term usage (somethink like 4bsd's load avg).
 * there are also averages for total IO, read, write, hits, and reclaims.
 * an average is increased for each reference (read/write),
 * and all averages are decreased (by a percentage) every 100 accesses.
 * the reference function is:    avg += A_INC; A_INC = 16;
 * the update function is:  avg -= 1/8*avg;
 * for these parameters, the total avg will tend between 12800 and 14400.
 */

#include <osbind.h>
#include "cache.h"

/* cache.c can compile with DLIBS or MWC */
#ifndef  MWC
#define  DLIBS
#endif

#ifdef MJC
#define  ASM  1    /* inline assembler */
#endif

#define  reg  register

#ifdef OK
typedef  long (*Func)();        /* function returning long */
#else
typedef  char *   Func;         /* good enough */
#endif

/* define system variables */
#define  bios_rw  (*(Func*)0x000476) /* bios read/write routine */
#define  bios_mc  (*(Func*)0x00047E) /* bios media change routine */
```

```
#ifdef OK
Func old_rw;
Func cur_mc;
#else
long (*old_rw)();       /* previous rwabs routine */
long (*cur_mc)();       /* current mediach routine */
#endif

/* per-device data */
int  c_dev = 0;         /* cached device, A: */
int  disable;      /* flag to disable */
short    stats [STATS];         /* global averages */
Cache    cache [NSECT];         /* per sector averages and cbufs indices */
short    csize;             /* size of cache in sectors */
Sector*  cbufs;             /* cached sector buffers */
short    cycle;             /* count from 100 to 0 */
short    nfree;             /* next free cache sector */

#ifdef DLIBS

extern char *_base, *_break;     /* program basepage, heap break */
long _STKSIZ = -2L;         /* grab all free memory for heap */

#else    /* MWC */

#include <basepage.h>
#define  _base    ((char *)BP)
#define  _break   _stack        /* MWC combines _STKSIZ and _break as _stack */
char   *_break = (char *)300*1024L; /* grab 300K for heap */

#endif

char header [] = "ST Floppy Disk Cache (by Eric Gisin)\r\n";

/*
 * setup bios disk IO routines, terminate and stay resident.
 */
main(argc, argv)
     int argc;
     char **argv;
{
     long stack;
     extern long cache_rw();
     extern char *sbrk(/*size_t n*/);

     Cconws(header);

     if (argc > 1) {
          char *arg = argv[1];
          if ('a' <= *arg && *arg <= 'z')
               c_dev = *arg++ - 'a';
          if ('A' <= *arg && *arg <= 'Z')
               c_dev = *arg++ - 'A';
```

```
          csize = atoi(arg) * 1024/sizeof(Sector);
     }
     if (csize < 100)
          csize = 100;       /* 50K is minimum */

#ifndef DEBUG
     if (Rwabs(CSTAT, (char*)NULL, 0, 0, c_dev) != NULL) { /* already loaded */
          Cconws("Cache already loaded\r\n");
          return 0;
     }
#endif

#if 0
     cbufs = (Sector*) sbrk(csize * sizeof(Sector));
#else                   /* avoid loading long multiply */
     cbufs = (Sector*) sbrk((size_t)&((Sector*)NULL)[csize]);
#endif
     if (cbufs == NULL) {
          Cconws("Not enough memory\r\n");
          Pterm(-1);
     }

     /* set up our routines for bios rwabs */
     stack = Super(NULL);        /* super mode */
     old_rw = bios_rw;
     bios_rw = cache_rw;
     cur_mc = bios_mc;
     Super(stack);          /* user mode */

     cache_init();

     Ptermres(_break - _base, 0);     /* terminate, stay resident */
}

/* clear the cache */
cache_init()
{
  reg    int  i;

     cycle = CYCLE;
     nfree = 0;
     for (i = 0; i < STATS; i ++)
          stats[i] = 0;
     for (i = 0; i < NSECT; i ++) {
          cache[i].avg = 0;
          cache[i].buf = NOBUF;
     }
     return 0;
}

/* our BIOS rwabs routine */
long
cache_rw(func, addr, count, sect, dev)
```

```
    int  func;          /* IO function */
    Sector*  addr;          /* buffer address */
    int  count;         /* sector count */
    int  sect;          /* sector number */
    int  dev;      /* disk drive */
{
  reg    int  s;
  reg    int  i;
    int  n;
    long l;

    if (dev != c_dev || disable)
        return (*old_rw)(func, addr, count, sect, dev);

    /* recognize magic function for cstat */
    if (func == CSTAT)
        return (sect==0) ? stats : cache;

    if (func == CINIT)
        return cache_init();

    /* check for media change */
    /* floppy rwabs will return -14 if real media change */
    if ((*cur_mc)(dev) != 0) {
        l = (*old_rw)(0, 0xFC0000, 0, 0, dev);    /* dummy read */
        if (l != 0) {
            cache_init();
            return l; /* always -14 (media changed)? */
        }
    }

    if (sect < 0 || count < 0 || sect+count < 0)
        return -5;          /* sector out of range */

    /* update cache averages */
    for (i = 0; i < count; i ++) {
        s = sect + i;
        stats[TOTAL] += A_INC;
        stats[((func&1)==0) ? READS : WRITES] += A_INC;
        if (s < NSECT)
            cache[s].avg += A_INC;
        if (-- cycle < 0) {
            cycle = CYCLE;
            update_averages();
        }
    }

    /* copy cache to read buffer */
    if ((func&1)==0 && sect>0) {
        for (i = 0; i < count; i ++) {
        s = sect + i;
        n = cache[s].buf;
        if (n == NOBUF)
```

```
            break;
        copy_sector(&cbufs[n], addr++);
        stats[HITS] += A_INC;
       }
       sect += i;     count -= i;
    }

    /* do BIOS IO */
    l = (*old_rw)(func, addr, count, sect, dev);
    if (l != 0)
        return l;

    /* update cache from IO buffer */
    for (i = 0; i < count; i ++) {
        s = sect + i;
        n = cache[s].buf;
        if (n == NOBUF)
            get_free(s);
        n = cache[s].buf;
        if (n != NOBUF)
            copy_sector(addr + i, &cbufs[n]);
    }

    return 0;
}

/*
 * update averages.
 * this is called once per N accesses.
 */
update_averages()
{
  reg    int  i;
    int  n;

    for (i = 0; i < STATS; ++ i) {
        stats[i] -= stats[i]+7 >> 3;
    }

    for (i = 0; i < NSECT; ++ i) {
        cache[i].avg -= cache[i].avg+7 >> 3;
        n = cache[i].buf;
        if (!(n == NOBUF || 0<=n && n<csize)) {
            disable = 1;
            Cconws("<cache: internal error>");
        }
    }
}

/*
 * get cache block with lowest avg less than requested block.
 * somewhat inefficient, but should not be called often.
 */
```

```
get_free(c)
     int  c;            /* sector */
{
  reg    int  s;
  reg    Cache *  p;
  reg    Cache *  m;

     if (nfree < csize) {
          cache[c].buf = nfree++;
          return;
     }

     /* find sector with a buffer and lowest avg */
     p = &cache[0];
     m = &cache[c];
     for (s = 0; s < NSECT; s ++, p ++)
          if (p->buf != NOBUF && p->avg <= m->avg)
               m = p;

     /* if that sector has lower avg than requested sector, reclaim */
     if (m->avg < cache[c].avg) {
          cache[c].buf = m->buf;
          m->buf = NOBUF;
          stats[RECLMS] += A_INC;
     }
}

copy_sector(s, d)
     Sector *s;
     Sector *d;
{
     int  i;
  reg    char   *dp = d->data;  /* A5 in MJC */
  reg    char   *sp = s->data;  /* A4 in MJC */

     if ((s&1) | (d&1)) {   /* odd address! */
          i = sizeof(Sector);
          while (--i >= 0)
               *dp++ = *sp++;
     } else {
#ifndef  ASM
          /* usually generates loop: move.w (A0)+, (A1)+; dbra D0, loop */
          *d = *s;
#else
          /* this is twice as fast as above */
          sizeof(Sector)/sizeof(long)/4 - 1;   /* result to D0 */
          /* isn't machine language fun!     loop: */
          asm(= 10972);          /* move.l (A4)+, (A5)+ */
          asm(= 10972);          /* move.l (A4)+, (A5)+ */
          asm(= 10972);          /* move.l (A4)+, (A5)+ */
          asm(= 10972);          /* move.l (A4)+, (A5)+ */
          asm(= 20936 = -10);         /* dbra D0, loop */
#endif
```

```
     }
}

/* allocate n bytes from heap (stack segment). NULL on error */
char *sbrk(n)
     size_t n;
{
     extern char *_break;        /* current heap break */
     char *p = _break;
     int dummy;             /* &dummy is almost SP */

     _break += n;
     if (_break+256 > (char*)&dummy) {
          _break = p;
          return NULL;
     }
     return p;
}

#if 0
int brk(p)
     char *p;           /* new break address */
{
     int dummy;             /* &dummy is almost SP */
     extern char *_break;        /* current break address */
     extern char *end;      /* initial _break; */

     if (p < end || p+256 > (char *)&dummy)
          return -1;
     _break = p;
     return 0;
}
#endif
@\Rogue\Monster\
else
  echo "shar: Will not over write cache.c"
fi
if `test ! -s cstat.c`
then
echo "x - cstat.c"
cat > cstat.c << '@\Rogue\Monster\'
/*
 * Disk Cache statistics
 */

#include <osbind.h>
#include "cache.h"

#define  reg  register

int  verbose = 0;
extern   char *   c1_4f();
```

```
main(argc, argv)
     char **  argv;
{
  reg    int  i = 1, n;
  reg    short *  stats;
  reg    Cache *  cache;
     int  dev = 0
     int  total;

     if (i < argc && argv[i][0]=='-') {
          verbose = 1;
          i ++;
     }
     if (i < argc)
          if ('a' <= argv[i][0] && argv[i][0] <= 'z')
               dev = argv[i][0] - 'a';

     stats = Rwabs(CSTAT, 0L, 0, 0, dev);
     cache = Rwabs(CSTAT, 0L, 0, 1, dev);
     if (stats == NULL || cache == NULL) {
          printf("Cache not installed\n");
          return 1;
     }

     total = stats[0];
     if (total==0) {
          printf("No statistics yet\n");
          return 1;
     }

     printf("Cache hits %s, ",   c1_4f(stats[HITS], total));
     printf("Disk reads %s, ",   c1_4f(stats[READS]-stats[HITS], total));
     printf("Disk writes %s; ",  c1_4f(stats[WRITES], total));
     printf("Buffer reclaims%s\n",    c1_4f(stats[RECLMS], total));

     if (!verbose)
          return 0;

     n = 0;
     for (i = 0; i < NSECT; i ++) {
          short    avg, buf;
          avg = cache[i].avg;
          buf = cache[i].buf;
          if (avg > 0 || buf != NOBUF) {
              printf("%4d%c%5s  ",
               i, (buf == NOBUF) ? '.' : ':', c1_4f(avg, total));
              if (++n % 6 == 0)
               printf("\n");
          }
     }
     printf("\n");
     return 0;
}
```

```
/* convert a/b in %1.3f format */
char * c1_4f(a, b)
     int a, b;
{
     register int i;
     register long l;
     register char * p;
     static char buf [12];

     p = buf;
     l = a;
     i = a/b;
     *p++ = (0<i && i<=9) ? '0'+i : ' ';
     *p++ = '.';
     for (i = 1; i <= 4; i++) {
          l = 10*(l%b);
          *p++ = '0' + (short)(l/b);
     }
     *p = '\0';
     return buf;
}

@\Rogue\Monster\
else
  echo "shar: Will not over write cstat.c"
fi
if `test ! -s cache.h`
then
echo "x - cache.h"
cat > cache.h << '@\Rogue\Monster\'
/*
 * Floppy Disk Cache (Atari ST)
 */

#ifndef NULL            /* belong in <stddef.h> */
#define  NULL 0L
typedef  long size_t;
#endif

/* magic functions to return internal tables to cstat */
#define  CSTAT    -1            /* return address of cache tables */
#define  CINIT    -2            /* clear cache */

/* cache parameters */
#define  NSECT    (82*2*10)     /* maximum sectors cached */
#define  CYCLE    100           /* update interval */

#define  A_INC    16            /* increment for each reference */

/* indices for stats array */
#define  TOTAL    0
#define  HITS 1
```

```
#define  READS    2
#define  WRITES   3
#define  RECLMS   4
#define  STATS    5             /* elements in stats */

#define  NOBUF    (-1)          /* no cached sector */

/* per sector cache information */
typedef struct {
     short    avg;      /* average usage */
     short    buf;      /* index of cached sector in cbufs [] */
} Cache;

typedef struct {
     char data [512];
} Sector;

@\Rogue\Monster\
else
  echo "shar: Will not over write cache.h"
fi
if `test ! -s readme`
then
echo "x - readme"
cat > readme << '@\Rogue\Monster\'
              Floppy Disk Cache

     What is a disk cache?
A disk cache is resident program that speeds up disk IO
by keeping a copy of recently used sectors of the disk
in RAM memory.  It is similar to a RAM disk but has the
advantage that files are always up-to-date on the disk.
You can't lose files like a RAM disk does when the system crashes.

This cache will typically eliminate from 50-85% of disk IO if enough
cached sectors are installed. There is a slight CPU
overhead in copying sectors to the cache after disk IO.

     Installation
The programs cache.ttp and cstat.ttp should be placed
in your directory for executable files, usually \bin.
cache.ttp should be run from a shell startup file;
for example gulam.g under Gulam or GemBoot's startup file.
The first argument to "cache" is the size of the cache in K-bytes,
which defaults to 50K.  The size can be preceeded by a drive letter.
Run "cache" twice if you need to cache two drives.
If you install cache.ttp as \auto\cache.prg
there is no way to set the drive or cache size without modifing the source.

     Operation
The cache clears automatically whenever you change disks.
Make sure all your disks have been formatted correctly with unique
serial numbers. An early version of twister and any bit-image
```

disk copy program will not generate unique serial numbers.
An easy way to check for unique serial numbers is:
list the root directory, switch disks, list the root directory.
If the directory lists are identical then the serial numbers may be the same.
You will probably trash disks even without a disk cache
when you have non-unique serial numbers.

The utility "cstat" will print some statistics averaged over
the last 1000 (approx.) sectors of IO.

     Other Hints
Using a "fast" formatter like DC Format will speed up disk IO further.
You can also turn off write verify with DC Format or pokew in Gulam,
but there is some risk of losing files then.
It also helps to create all directories before anything else
is placed on a new disk, this speeds up file creation and deletion.

     Compilation
This program is written completly in C for readability and portablity.
The cache program should be compiled with the dLibs library or MWC.
It requires the _base, _break, and _STKSIZ variables dLibs implements.
Non-standard compilers may require a "define short int".
Please indicate in the source and object when you make non-trivial changes.

     Possible Extensions
Modify it for multiple hard disk partitions. This is non-trivial.
The current algorithm assumes sectors can be accessed through
cache[sector]. The cache array would be 160K bytes for a 20M byte drive,
and is scanned sequentially in a couple of places.

     Eric Gisin, egisin@UWaterloo.CA
@\Rogue\Monster\
else
  echo "shar: Will not over write readme"
fi
to concatenate archives, remove anything after this line
exit 0

From 488616556889182176
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,f2249d02580d7071,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-20 09:18:57 PST
Path: utzoo!attcan!uunet!husc6!think!ames!pasteur!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i033: vdit -- VDI interface with GDOS font support
Message-ID: <3363@saturn.ucsc.edu>
Date: 20 May 88 16:18:57 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 556
Approved: koreth@ssyx.ucsc.edu

Keywords:

Submitted-by: egisin@watmath.waterloo.edu (Eric Gisin)
Posting-number: Volume 1, Issue 33
Archive-name: vdit

This is a partial PD VDI interface with support for loading GDOS fonts.
See the file "readme" at the beginning of the shell archive.
A set of PD GDOS fonts has been posted to comp.binaries.atari.st.

[I don't know which C compiler was used to develop this program, but the
 code looks fairly non-compiler-specific. -sg]

```
# This is a shell archive.  Remove anything before this line,
# then unpack it by saving it in a file and typing "sh file".
#
# Wrapped by watmath!egisin on Tue May 17 11:30:45 EDT 1988
# Contents:  readme gfont.h loadfont.c vdi.c vdi.h vdit.c

echo x - readme
sed 's/^@//' > "readme" <<'@//E*O*F readme//'
This package contains a partial public domain VDI interface.
It includes a working interface to the undocumented
VDI "load_fonts" and "unload_fonts" functions,
which allow you to use other fixed and proportional fonts with GEM's VDI.
See vdit.c for documentation and an example.
Most of vdi.h/vdi.c is untested, except for the functions
used by vdit.c and by my VT200 emulator.
If you make any fixes or enhancements to the VDI I'd like to see them.
I have also posted a public domain fixed width font family
suitable for use with terminal emulators to comp.binaries.atari.st.

vdi.h:        structure and macro definitions for VDI
vdi.c:        interface functions for VDI
gfont.h: definition of a GDOS font
loadfont.c:   load GDOS fonts from disk
vdit.c:       a simple program to load fonts and display them

     Eric Gisin, egisin@math.UWaterloo.CA
@//E*O*F readme//
chmod u=rw,g=r,o=r readme

echo x - gfont.h
sed 's/^@//' > "gfont.h" <<'@//E*O*F gfont.h//'
/*
 * Internal representation of GDOS font. Author unknown.
 */

typedef struct {
     short    font_id;  /* Font face identifier, 1 -> system font */
     short    size;          /* Font size in points */
     char name[32];/* Font name */
     short    first_ade;    /* Lowest ADE value in the face */
```

```
     short    last_ade;/* Highest ADE value in the face */
     short    top;     /* Distance of top line relative to baseline */
     short    ascent;       /* Distance of ascent line relative to baseline */
     short    half;         /* Distance of half line relative to baseline */
     short    descent; /* Distance of decent line relative to baseline */
     short    bottom;       /* Distance of bottom line relative to baseline */
                   /* All distances are measured in absolute values */
                   /* rather than as offsets. They are always +ve */
     short    max_char_width;    /* Width of the widest character in font */
     short    max_cell_width;    /* Width of the widest cell character cell in
face */
     short    loffset; /* Left Offset see Vdi appendix G */
     short    roffset; /* Right offset   "      "     " */
     short    thicken; /* Number of pixels by which to thicken characters */
     short    ul_size; /* Width in pixels of the underline */
     short    lighten; /* The mask used to lighten characters */
     short    skew;         /* The mask used to determine when to perform */
              /* additional rotation on the character to perform skewing */
     short    flags;        /* Flags */
                   /*  $01 default system font */
                   /*   02 horiz offset table should be used */
                   /*   04 byte-swap flag (thanks to Motorla idiots) */
                   /*   08 mono spaced font */
     short *  hoff_base;    /* Offset to horizontal offset table */
     short *  coff_base;    /* Offset to character offset table */
          /* font bitmap is byte[width][height], width must be even */
     short *  form_base;    /* Offset to font data */
     short    form_width;   /* Form width (#of bytes/scanline in font data) */
     short    form_height;  /* Form height (#of scanlines in font data) */
     char *   next_font;    /* Pointer to next font in face */
} GFont;

@//E*O*F gfont.h//
chmod u=rw,g=r,o=r gfont.h

echo x - loadfont.c
sed 's/^@//' > "loadfont.c" <<'@//E*O*F loadfont.c//'
/*
 * Load GDOS font, Eric Gisin.
 * loads a font from file name.FED. Tries in ., then $FONTDIR, then A:\FONT.
 * does not have enough error checking.
 */

#define  NULL 0L
#include <gfont.h>

#define  open(file, mode)   gemdos(0x3D, file,mode)
#define close(fd)       gemdos(0x3E, fd)
#define  read(fd, buf, size)    gemdos(0x3F, fd,(long)size,buf)

#define  Swapw(v)  (v) = ((v)<<8) | ((v)>>8)&0x00FF
#define  Swapl(v)  (v) = ((v)>>24)&0x000000FF | ((v)>>8)&0x0000FF00
```

```
char * font_dir = NULL;
extern char * malloc();
extern char * getenv();

/*
 * load GDOS font, return pointer to font.
 */
GFont *
load_font(name)
     char *   name;
{
     int fd, i, n, size;
     char temp [64], * p = temp;
     char * q, * tcwd;
     int swapf;
     register GFont*    fp;

     if (font_dir == NULL)
          font_dir = getenv("FONTDIR");
     if (font_dir == NULL)
          font_dir = "A:\\font";

     fp = malloc(sizeof(GFont));
     if (fp == NULL) return NULL;

     /* set up font file's path name */
     for (q = font_dir; *q; )
          *p++ = *q++;
     *p++ = '\\';
     tcwd = p;
     for (q = name; *q; )
          *p++ = *q++;
     for (q = ".fed"; *q; )
          *p++ = *q++;
     *p = '\0';

     /* open font file */
     fd = open(tcwd, 0);         /* try name.fed */
     if (fd < 0)
          fd = open(temp, 0);
     if (fd < 0)
          return (0);

     /* read header */
     size = sizeof(GFont);
     read(fd, fp, size);         /* should (char *)fp */

     swapf = !(fp->flags&0x0004);
     if (swapf) {
          /* swap header words and longs */
          Swapw(fp->font_id);         Swapw(fp->size);
          Swapw(fp->first_ade);       Swapw(fp->last_ade);
          Swapw(fp->top);             Swapw(fp->ascent);
```

```
          Swapw(fp->half);       Swapw(fp->descent);
          Swapw(fp->bottom);
          Swapw(fp->max_char_width);  Swapw(fp->max_cell_width);
          Swapw(fp->loffset);         Swapw(fp->roffset);
          Swapw(fp->thicken);         Swapw(fp->ul_size);
          /* Swapw(fp->flags); */     Swapl(fp->hoff_base);
          Swapl(fp->coff_base);       Swapl(fp->form_base);
          Swapw(fp->form_width);      Swapw(fp->form_height);
     }

     /* read offset table */
     n = fp->last_ade + 1 - fp->first_ade + 1;
     size = n * sizeof(short);
     fp->coff_base = malloc(size);
     if (fp->coff_base == NULL) return NULL;
#if 0
     for (i = 0; i < fp->first_ade; i++)
          fp->coff_base[i] = 0;
#endif
     read(fd, fp->coff_base, size);
     if (swapf)         /* swap offset array words */
          for (i = 0; i < n; i++)
               Swapw(fp->coff_base[i]);

     /* read bitmap */
     size = fp->form_width*fp->form_height;
     fp->form_base = malloc(size);
     if (fp->form_base == NULL) return NULL;
     read(fd, fp->form_base, size);

     close(fd);
     return fp;
}

/* load a set of fonts, linking them for vst_load_font */
GFont *
load_fonts(fontv)
     char **fontv;
{
     int errors = 0;             /* should return */
     GFont * fhead = NULL;
     GFont * f, ** fpp = &fhead;

     /* load fonts, create linked list fhead */
     while (*fontv != NULL) {
          *fpp = f = load_font(*fontv++);
          if (f != NULL)
               fpp = &f->next_font;
          else
               errors++;
     }
     return fhead;
}
```

```
@//E*O*F loadfont.c//
chmod u=rw,g=r,o=r loadfont.c

echo x - vdi.c
sed 's/^@//' > "vdi.c" <<'@//E*O*F vdi.c//'
/*
 * PD mini VDI interface, Eric Gisin
 */

#include <osbind.h>
#include <vdi.h>
#include <gfont.h>

#define  NULL 0L

static   short ct [12];         /* VDI/AES control block */
static   short ii [140];
static   short pi [16];
static   short io [47];
static   short po [16];
struct Params v_ps = {ct, ii, pi, io, po, NULL};
struct Params v_pt = {ct, ii, pi, io, po, NULL}; /* modifiable version of v_ps */

/* VDI operations with no parameters */
vdi_none(h, op)
     short h, op;
{
     vdi_setup(op, h, 0, 0);
     vdi(&v_ps);
}

/* VDI operations with one integer parameter */
vdi_int1(h, op, i)
     short h, op;
     short i;
{
     ii[0] = i;
     vdi_setup(op, h, 1, 0);
     vdi(&v_ps);
     return io[0];
}

/* VDI operations with one point parameter */
vdi_pnt1(h, op, x, y)
     short h, op;
     short x, y;
{
     pi[0] = x, pi[1] = y;
     vdi_setup(op, h, 0, 1);
     vdi(&v_ps);
}
```

```
vdi_pnt4v(h, op, fn, pnt)
     short h;
     short op, fn;
     short *pnt;
{
     register int i;

     for (i = 0; i < 4; i++)
          pi[i] = pnt[i];
     vdi_setup(op, h, 0, 2);
     ct[5] = fn;
     vdi(&v_ps);
}

/* open virtual work */
v_opnvwk(in, hp, out) /* todo: vdi_open */
     short in[];
     short *hp;
     short out[];
{
     v_pt.ii = in;
     v_pt.io = out;
     v_pt.pi = pi;
     v_pt.po = out+34;
     vdi_setup(100, *hp, 11, 0);
     vdi(&v_pt);
     *hp = ct[6];
}

vdi_cpyfm(h, op, mode, sdr, src, dst)
     short h, op;
     short mode;
     short *sdr;            /* source/dest rectangles */
     Form *src, *dst;       /* source/dest forms */
{
     register int i;

     *((Form**)&ct[7]) = src;
     *((Form**)&ct[9]) = dst;
     ii[0] = mode;
     for (i = 0; i < 8; i++)
          pi[i] = sdr[i];
/***
     *((Rect*)&pi[0]) = sdr[0];
     *((Rect*)&pi[4]) = sdr[1];
***/
     vdi_setup(op, h, 1, 4); vdi(&v_ps);
}

vst_alignment(h, x, y, xp, yp)
     short h;
     short x, y;
```

```
     short *xp, *yp;
{
     ii[0] = x, ii[1] = y;
     vdi_setup(39, h, 2, 0);
     vdi(&v_ps);
     *xp = io[0], *yp = io[1];
}

v_gtext(h, x, y, s)
     short h;
     short x, y;
     unsigned char *s;
{
     register int i;

     for (i = 0; *s != 0 && i < 140; i++)
          ii[i] = *s++;
     pi[0] = x; pi[1] = y;
     vdi_setup(8, h, i, 1);
     vdi(&v_ps);
}

vqt_attributes(h, at)
     short *at;    /*struct vdi_tattr *at;*/
{
     v_pt.io = at; v_pt.po = at+6;
     vdi_setup(38, h, 0, 0);
     vdi(&v_pt);
}

static char *vdi_scratch = NULL; /* VDI/LineA scratch buffer */

#if 0
/* load GDOS fonts using user's vx_load_fonts */
vst_load_fonts(h, select)
     short h, select;
{
     GFont * fp;
     extern char * malloc();
     extern GFont * vx_load_fonts();  /* user routine to load fonts */

     if (vdi_scratch == NULL)
          vdi_scratch = malloc(1024); /* scratch buffer */
     *((char **)&ct[7]) = vdi_scratch;
     ct[9] = 256;           /* offset of enlargment buffer */
     *((char **)&ct[10]) = fp = vx_load_fonts(select); /* GDOS font list */
     if (fp == 0)
          return 0;
     vdi_setup(119, h, 0, 0);
     vdi(&v_ps);
     return io[0];          /* number of fonts loaded */
}
```

```
vst_unload_fonts(h)
{
     extern void vx_unload_fonts();

     vdi_setup(120, h, 0, 0);
     vdi(&v_ps);
     vx_unload_fonts();
}
#endif

/* load GDOS fonts using fp */
/* make sure your ct array is large enough if you port this */
vstx_load_fonts(h, fp)
     short h;
     GFont *fp;
{
     extern char * malloc();

     if (vdi_scratch == NULL)
          vdi_scratch = malloc(1024); /* scratch buffer */
     *((char **)&ct[7]) = vdi_scratch;
     ct[9] = 256;            /* offset of enlargment buffer */
     *((char **)&ct[10]) = fp;
     if (fp == NULL)
          return 0;
     vdi_setup(119, h, 0, 0);
     vdi(&v_ps);
     return io[0];           /* number of fonts loaded */
}

vstx_unload_fonts(h)
{
     vdi_setup(120, h, 0, 0);
     vdi(&v_ps);
}

#if 0
/*
 * This code invokes VDI ESC 102 code (page 440 in the July '86
 * Abacus ST Internals BIOS listing).  I decompiled some library
 * routines to see how to make it work.  It may not work on new ROMS.
 * Martin Minow, Arlington MA 02174.  This routine is in the public domain.
 */
vt_set_font(fp)
     register GFont * fp;
{
     (*(GFont**)&ii[0]) = fp;    /* Put font info in intin[0] */
     vdi_setup(5, 0, 2, 0);
     ct[5] = 102;
     vdi(&v_ps);
}
#endif
```

```
@//E*O*F vdi.c//
chmod u=rw,g=r,o=r vdi.c

echo x - vdi.h
sed 's/^@//' > "vdi.h" <<'@//E*O*F vdi.h//'
/*
 * PD mini VDI interface, Eric Gisin
 */

/* Memory Form descriptor */
typedef struct {
     short *  base;
     short    width;
     short    height;
     short    wwidth;
     short    standard;
     short    planes;
     short    resv1, resv2, resv3;
} Form;

/* Rectangle descriptor */
typedef struct {
     short    x1, y1;       /* upper left */
     short    x2, y2;       /* lower right */
} Rect;

/* VDI/AES parameter block */
struct Params {
     short  *ct;            /* control */
     short  *ii;            /* ints in */
     short  *pi;            /* points in */
     short  *io;            /* ints out */
     short  *po;            /* points out */
     short  *xx;
};

#define  vdi(params)        bdos(115, params)
#define  vdi_setup(f, h, ii, pi) {ct[0] = f; ct[6] = h; ct[1] = pi; ct[3] = ii; }

#define  v_clsvwk(h)        vdi_none(h, 101)
#define  v_clrwk(h)         vdi_none(h, 3)
#define  vro_cpyfm(h, m, r, s, d) vdi_cpyfm(h, 109, m, r, s, d)
#define  vrt_cpyfm(h, m, r, s, d) vdi_cpyfm(h, 121, m, r, s, d)
#define  vsf_color(h, c)         vdi_int1(h, 25, c)
#define  vsf_perimeter(h, f)     vdi_int1(h, 104, f)
#define  vsf_interior(h, s) vdi_int1(h, 23, s)
#define  vsf_style(h, s)         vdi_int1(h, 24, s)
#define  vst_color(h, c)         vdi_int1(h, 22, c)
#define  vst_effects(h, e)  vdi_int1(h, 106, e)
#define  vst_point(h, s)         vdi_int1(h, 107, s)
#define  vst_rotation(h, a) vdi_int1(h, 13, a)
#define  vst_font(h, f)          vdi_int1(h, 21, f)
#define  vswr_mode(h, m)         vdi_int1(h, 32, m)
```

```
#define  vst_height(h, ht)  vdi_pnt1(h, 12, 0, ht)
#define  v_bar(h, p)        vdi_pnt4v(h, 11, 1, p)
#define  v_rbox(h, p)       vdi_pnt4v(h, 11, 8, p)
#define  v_rfbox(h, p)      vdi_pnt4v(h, 11, 9, p)
#define  v_recfl(h, p)      vdi_pnt4v(h, 114, 0, p)

@//E*O*F vdi.h//
chmod u=rw,g=r,o=r vdi.h

echo x - vdit.c
sed 's/^@//' > "vdit.c" <<'@//E*O*F vdit.c//'
/*
 * Sample program to load fonts, display them. Eric Gisin.
 * try: vdit -f91 fixed8 fixed9 fixed10 fixed11 fixed12
 *
 *   void vstx_load_fonts(short vdi_handle, GFont *fonts);
 *   void vstx_unload_fonts(short vdi_handle);
 *
 * These two undocumented VDI functions are used to add fonts
 * to a virtual screen. Unlike the VDI function vst_load_fonts,
 * which requires GDOS, it does not load the fonts from disk.
 * Instead you load them explicitly with load_fonts (or load_font),
 * and pass the linked list of fonts to vstx_load_fonts as "fonts".
 * To change the fonts in a virtual screen, unload then load again.
 *
 * The fonts must have valid font_id and size information.
 * The font_id is used by vst_font to select a font family.
 */

#include <vdi.h>
#include <gfont.h>

#define  NULL 0L

extern   GFont *load_fonts();

main(argc, argv)
     char ** argv;
{
     register int i;
     int font = 1, size = 10;
     short h = 1;
     short in [11];
     short out [57];

     /* parse options */
     for (i = 1; i < argc; i ++) {
          register char * arg = argv[i];
          if (arg[0]=='-') {
              if (arg[1]=='f')
               font = atoi(&arg[2]);
              if (arg[1]=='p')
               size = atoi(&arg[2]);
```

```
          } else
               break;
     }

     /* set up opnvwk parameters */
     in[0] = 1;             /* device is screen */
     in[1] = in[2] = 1;     /* line type/color */
     in[3] = in[4] = 1;     /* marker type/color */
     in[5] = in[6] = 1;     /* text font/color */
     in[7] = in[8] = in[9] = 1;  /* fill type/style/color */
     in[10] = 2;            /* raster coordinates */

     v_opnvwk(in, &h, out);
     vstx_load_fonts(h, load_fonts(&argv[i]));
     v_clrwk(h);
     i = vst_font(h, font);
     for (i = 6; i < 25; i += (i<14) ? 1 : 2) {
          vst_point(h, i);
          v_gtext(h, 0, i*16, "A test of load_fonts ?! {123<@>}");
     }
     vstx_unload_fonts(h);
     v_clsvwk(h);
     return 0;
}

@//E*O*F vdit.c//
chmod u=rw,g=r,o=r vdit.c

exit 0


From -1556153798061210027
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,759cdd69b60d6c4a,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-24 16:07:38 PST
Path: utzoo!utgpu!water!watmath!clyde!att!osu-cis!tut.cis.ohio-state.edu!bloom-
beacon!mit-eddie!ll-xn!ames!pasteur!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i034: flip -- disk copier, part01/02
Keywords: Megamax
Message-ID: <3412@saturn.ucsc.edu>
Date: 24 May 88 23:07:38 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1011
Approved: koreth@ssyx.ucsc.edu

Submitted-by: uunet!mcvax!mirsa.inria.fr!colas (Colas Nahaboo)
Posting-number: Volume 1, Issue 34
Archive-name: flip/part01

[This allows you to format and use the backs of single-sided disks, among
```

other things. Executable is being posted to the binaries group. -sg]

```
#-------------------------Cut Here-------------------------
#! /bin/sh
# This is a shell archive.  Remove anything before the "#! /bin/sh" line,
# then unpack it by saving it in a file and typing "sh file."
#
# Wrapped by Steven Grimm (koreth) at ssyx on Tue May 24 15:54:23 1988
#
# unpacks with default permissions
#
# Contents : FLIP.C
#
if `test ! -s FLIP.C`
then
echo "x - FLIP.C"
cat > FLIP.C << '@\Rogue\Monster\'
/*****************************************\
*                           *
* flip: Unprotected disk copier           *
* By Colas NAHABOO (INRIA sophia-antipolis)  *
*                           *
* Compile with MEGAMAX C v1.1 (in-line asm)  *
*                           *
\*****************************************/

#define VERSION "2.6b"
#define DATE  "Mar 23 88"

char           *header =
"\33E\33e\33v\033p\
FLIP: disk copier by Colas Nahaboo     v%-5s %-9s           Free Ram:%4ldK\
\033q\n\n";

#include <osbind.h>
#include <stdio.h>

#define BLANK (0xE5E5)      /* std sector format value */
#define MAGIC (0x87654321L) /* magic number value */
#define MEMORY_LEFT 0       /* bytes not eaten by main buffer */
#define TRACK_RETRY 4       /* number of full track retrys */
#define NO_DISK        0    /* no disk in drive */
#define SOURCE_DISK    1    /* source disk in drive */
#define DEST_DISK  2    /* destination disk in drive */

                  /* list of fatal errors */
#define NORMAL         10
#define USER_ABORT 1
#define SYNTAX_ERROR   2
#define NO_DISK_IN_MEMORY 3
#define NO_OVERRIDE    4
#define NO_DRIVE_B 5
#define DOUBLE_ON_BACK 6
```

```
#define NO_FLIP_ON_DOUBLE 7
#define BAD_OPTION 8
#define NO_MEMORY  9

int bad_option     =    ' ';

/* twister interface */
extern          twister();  /* adress of formatter */
char            buffer_data[0x3FFF]; /* 16k of buffer */
extern long     buffer;     /* format track buffer */
extern int      thedisk,    /* A=0, B=1 */
                dblsided,   /* SS=0, DS= -1 */
                sectoroffset,    /* 0 = sectors 0 to 10, 1= 11 to 20 */
                flipped,    /* normal=0, back side=1 */
          keypress,/* 1 if key aborted format */
                starttrack, /* 0 or more */
                endtrack;   /* 80 or more */
extern int      badflag;    /* return code */

/* flip's globals */
long            buffer_size = 0L;     /* max size of buffer */
jmp_buf         start_dest; /* insert dest */
int       drives_content[2];/* what is there in the drives ? */

int               /* disk parameters read from boot sect */
                drive,      /* drive: 0=A, 1=B */
                sides,      /* 0=SS, 1=DS */
                tracks,     /* 80 normally */
                st,         /* sectors per track (9 or 10) */
                bps;        /* bytes per sector (512) */

int               /* options */
          verify_source,     /* just verify the source */
          read_source,  /* read source */
          zero_disk,    /* prepares blank image in memory */
          write_dest,   /* write dest */
                format,     /* do format */
                Verify,     /* do verify formatting */
                verify,     /* do verify writing */
          override,/* do not read boot sector */
          single,       /* Single Sided */
          nine_sectors, /* 9 sect/track */
                interactive,     /* interactive */
                use_track_map;   /* to use built-in track map */

int             d1=0, d2=0, s1=0, s2=0; /* drive/side source/dest */

#define TRACK_MAP_SIZE  99
char            track_map[TRACK_MAP_SIZE + 1];
char            full_track_map[TRACK_MAP_SIZE + 1]; /* for multipasses */

int             one_valid_copy_done = 0;  /* used to allow "r" */
int             multipass;  /* when not enough mem */
```

```
int      first_pass,
         last_pass;

jmp_buf         start;
char           *buf = NULL;
char           *malloc(), *index(), *get_drive();
unsigned  is_non_executable();
extern char    *help_text[];
int             errno;

char            rep[40], default_rep[40]; /* input buffer */

/*
 * parse the command line and set options
 * return 1 if error, 0 if OK
 */

int
parse_command()
{
     printf("Command (? for help) (CR= %s ): ", default_rep);
     fflush(stdout);
     gets(rep);
     /* parse main options */
     if (!rep[0]) {          /* void rep ==> default */
         strcpy(rep, default_rep);
     }
     verify_source = read_source = zero_disk = write_dest = format =
     verify = Verify = use_track_map = override = nine_sectors = single =
     interactive = d1 = d2 = s1 = s2 = 0;
     switch(rep[0]){
     case '?':     /* need help */
         more(help_text);
         longjmp(start, NORMAL);
     case 'q':     /* quit */
         exit(0);
     case 'v':
         verify_source = 1;
         get_options(get_drive(SOURCE_DISK, rep+1, 1), "os9ti");
         return 0;
     case 'r':
         if (!one_valid_copy_done)
          longjmp(start, NO_DISK_IN_MEMORY);
         get_options(get_drive(DEST_DISK, rep+1,1), "fvVi");
         write_dest = 1;
         return 0;
     case 'f':
         override = 1;
         get_options(get_drive(DEST_DISK, rep+1,1), "s9tvVfi");
         zero_disk = 1;
         write_dest = 1;
         return 0;
     case 'a': case 'b':
```

```
        get_options(
         get_drive(DEST_DISK,
                get_drive(SOURCE_DISK, rep, 0),
                1),
         "os9tvVfi");
        read_source = write_dest = 1;
        return 0;
     default :
        longjmp(start, SYNTAX_ERROR);
     }
}

char *
get_drive(drive, string, is_optional)
int drive;
char * string;
int is_optional;
{
    char *ptr =string;
    int side = 0, default_drive = 0;

    if(!index("abAB", string[0]))
     if(is_optional)
        default_drive = 1;      /* drive A is default */
     else
        longjmp(start, SYNTAX_ERROR);
    if(!default_drive && (index("12", string[1]))) {
     ptr ++;
     side = (toupper(*ptr) == '2' ? 1 : 0);
    }
    if(drive == DEST_DISK){
     d2 = (default_drive ? 0 : (toupper(*string) == 'B' ? 1 : 0));
     s2 = side;
    }else{
     d1 = (default_drive ? 0 : (toupper(*string) == 'B' ? 1 : 0));
     s1 = side;
    }
    return (default_drive ? string : ptr +1);
}

get_options(string, options)
char *string, *options;
{
    while(*string){
     if(!index(options, *string)) {
        bad_option = *string;
        longjmp(start, BAD_OPTION);
     }
     switch(*string){
     case 'o': override = 1; break;
     case 's': single = 1; break;
     case '9': nine_sectors = 1; break;
     case 't': use_track_map = 1; break;
```

```
     case 'v': verify = 1; break;
     case 'V': Verify = 1; break;
     case 'f': format = 1; break;
     case 'i': interactive = 1; break;
     }
    string++;
    }
    if(!override && (single || nine_sector))
     longjmp(start, NO_OVERRIDE);
}

main()
{

    buffer = (long) buffer_data;/* init twister */
    sectoroffset = 0;       /* mem alloc */
    buffer_size = (long) Malloc(-1L) - MEMORY_LEFT;
    buf = (char *) Malloc(buffer_size);
    if(is_drive_B())        /* default command */
     strcpy(default_rep, "bfv");
    else
     strcpy(default_rep, "afv");
    printf(header, VERSION, DATE, buffer_size / 1024);  /* resets screen */
    switch(setjmp(start)){
    case 0:case NORMAL:          /* initial */
     break;
    case USER_ABORT:             /* user abort */
     printf("\007Ok, aborted...\n");
     break;
    case SYNTAX_ERROR:           /* syntax error */
     printf("\007SYNTAX ERROR!\n");
     break;
    case NO_DISK_IN_MEMORY:
     printf("\007No disk in memory!\n");
     break;
    case NO_OVERRIDE:
     printf("\007Options d and 9 requires option o\n");
     break;
    case NO_DRIVE_B:
     printf("\007Sorry, no drive B present...\n");
     break;
    case DOUBLE_ON_BACK:
     printf("\007A double sided disk on back? use override!\n");
     break;
    case NO_FLIP_ON_DOUBLE:
     printf("\007You cannot flip double-sided disks!\n");
     break;
    case BAD_OPTION:
     printf("\007bad option: %c\n", bad_option);
     break;
    case NO_MEMORY:
     printf("\007Sorry, not enough memory...\n");
     break;
```

```
    }
    while (1) {
     /* we dont know what's in the drives */
     drives_content[0] = drives_content[1] = NO_DISK;
     /* prompts and parse command and options */
     parse_command();
     /* check for drive B */
     if ((d1 || d2) && (!is_drive_B()))
         longjmp(start, NO_DRIVE_B);
     /* prompts for track_map */
     if (use_track_map)
         input_track_map();
     /* examine source disk */
     if(read_source || verify_source || zero_disk)
         examine_disk(d1, s1);
     /* check run time consistency */
     if (sides == 2) {
         if (s1 == 1)
             longjmp(start, DOUBLE_ON_BACK);
         if (s1 != s2)
          longjmp(start, NO_FLIP_ON_DOUBLE);
     }
     strcpy(default_rep, rep);   /* seems ok then take it as default */

     /* copy disk */
     if((d1 != d2) &&   /* suppose user had time to put dest */
          (read_source || verify_source))
         drives_content[d2] = DEST_DISK;
     do_copy_disk();
     printf("\007OK, done.\n");
    }
}

/*
 * do_copy_disk:
 * do a one-pass or multipass copy
 */

do_copy_disk()
{
     int             track_number = 0, i, j, to_do;

     multipass = 0;
     if (read_source) {
         for (i = 0; i < TRACK_MAP_SIZE; i++)
          track_number += (int) track_map[i];
         if(buffer_size < (sides * bps * st * (long) track_number))
             multipass = 1;
         one_valid_copy_done = !multipass; /* let previous value of
               multipass inchanged if no read_source */
     }
     if(zero_disk) one_valid_copy_done = 0;
     bcopy(track_map, full_track_map, TRACK_MAP_SIZE);
```

```
    if(!multipass){
        copy_disk();
    }else{
        for(j=0; j<TRACK_MAP_SIZE; j++)
         track_map[j] = '\0';
        track_number = buffer_size / (sides*bps*st);
        i = to_do = 0;
        first_pass = 1;
        last_pass  = 0;
        while(i < TRACK_MAP_SIZE){
         if(full_track_map[i]){
             to_do++;
             track_map[i] = '\001';
         }
         if(to_do == track_number){
             copy_disk();
             to_do = first_pass = 0;
             for(j=0; j<TRACK_MAP_SIZE; j++)
              track_map[j] = '\0';
         }
         i++;
        }
        first_pass = 0;
        last_pass = 1;
        copy_disk();
    }
}


/*
 * examine_disk: reads boot sector to determine type of disk.
 */

/* describe the BOOT sector by offsets in the boot sector */
#define b_bootbranch 0
#define b_oem 0x02
#define b_serial 0x08
#define b_bps 0x0B
#define b_spc 0x0D
#define b_res 0x0E
#define b_nfats 0x10
#define b_ndirs 0x11
#define b_nsects 0x13
#define b_media 0x15
#define b_spf 0x16
#define b_spt 0x18
#define b_nsides 0x1A
#define b_nhide 0x1C
#define b_boot 0x1E
#define b_checksum 0x01FE

/* 2 chars ==> 1 int */
#define canshort(x,y) (((unsigned)(x))|(((unsigned)(y))<<8))
```

```
unsigned char   bb[512];

randomize_boot()
{
    long serial;
    unsigned non_exec = is_non_executable(buf);

    if((!multipass || first_pass) && track_map[0]) {
     serial = Random();
     buf[b_serial] = (char) ((serial >> 16) & 255L);
     buf[b_serial+1] = (char) ((serial >> 8) & 255L);
     buf[b_serial+2] = (char) (serial & 255L);
     update_checksum(buf, non_exec);
    }
}

/*
 * prepares a blank image:
 * boot sector, & zero tracks 0&1 (if enough memory)
 */

do_zero_disk()
{
    unsigned nsects;

    if(buffer_size < (sides * bps * st * 2L)) longjmp(start, NO_MEMORY);
    bfill(buf, '\0', (long) (sides * bps * st * 2L));
    Protobt(buf, 0x02000000L, ((sides == 2) ? 3 : 2), 0);
    buf[b_spt] = (char) (st & 0xFF);
    nsects = (unsigned) (sides * st * (1 + max_track()));
    buf[b_nsects] = (char) (nsects & 255);
    buf[b_nsects+1] = (char) ((nsects >> 8) & 255);
    update_checksum(buf, (unsigned) 1);
    printf("Formatting %s sided disk, %d tracks and %d sectors.\n",
     ((sides == 1) ? "single" : "double"), 1+max_track(), st);
}

int
max_track()
{
    int i;

    for (i= TRACK_MAP_SIZE -1; i>=0; i--)
     if(full_track_map[i]) return i;
    return 0;
}

examine_disk(drive, side)
int             drive, side;
{
    int             i;
```

```
    if (override) {
     sides = (single ? 1 : 2);
     bps = 512;
     st = (nine_sectors ? 9 : 10);
     tracks = 80;
    } else {
     if (drives_content[drive] != SOURCE_DISK) {
         prompt_for_disk(SOURCE_DISK, drive, side);
     }
     while (Floprd(&bb, 0L, drive, 1, 0, side, 1)) {
         printf("\007Cannot read boot block! CR/SP=retry, other=abort: ");
         fflush(stdout);
         if (!index("\r\n ", (char) Cnecin()))
          longjmp(start, USER_ABORT);
     }
     if(!is_non_executable(bb)) {
    printf("\007EXECUTABLE BOOT! at $3A, %2x %2x %2x -- check for virus\n",
         (int) bb[58], (int) bb[59], (int) bb[60]);
     }
     printf("Disk in drive %c:", 'A' + drive);
     printf(" %d side", sides = (int) bb[b_nsides]);
     printf("%s, %d tracks,", (sides == 2 ? "s" : ""),
            tracks = ((canshort(bb[b_nsects], bb[b_nsects + 1]) /
                (int) bb[b_spt]) /
               (int) bb[b_nsides]));
     printf(" %d sectors of", st = (int) bb[b_spt]);
     printf(" %d bytes\n", bps =
            canshort(bb[b_bps], bb[b_bps + 1]));
     if ((bps != 512) || ((st != 10) && (st != 9)) || (tracks != 80)) {
         printf("\007WARNING -- this is not a normal disk!\n");
     }
     if (sides < 1 || sides > 2 ||
         bps != 512 ||
         st > 11 || st < 9 ||
         tracks < 1 || tracks > TRACK_MAP_SIZE) {
         printf("***** Strange disk! ... copying normally ");
         printf("(1 side, 80 tracks of 9 sectors)\n");
         override = 1;
         sides = 1;
         bps = 512;
         st = 9;
         tracks = 80;
     }
    }
    if (!use_track_map) {
     for (i = 0; i < tracks; i++)
         track_map[i] = '\001';
     for (i = tracks; i < TRACK_MAP_SIZE; i++)
         track_map[i] = '\0';
    }
}

/*
```

```
 * format disk using TWISTER
 * returns 0 if OK, aborts and returns 1 on error.
 */

int
format_disk(f_drive, f_double, f_side)
int             f_drive, f_double, f_side;
{
    int             i;
    char     *map;

    if(multipass)
        map = full_track_map;
    else
     map = track_map;
    thedisk = f_drive;
    dblsided = (f_double ? -1 : 0);
    flipped = f_side;
redo:
    i = 0;
    while (i < TRACK_MAP_SIZE) {
     if (map[i]) {
         starttrack = i;
         while (map[i++]);
         endtrack = i - 1;
         keypress = 0;
         Supexec(twister);
         if(keypress) {
          longjmp(start, USER_ABORT);
         }
         if (badflag){
          printf("\n\007FORMAT Error! CR/SP = retry whole formatting,");
          printf(" other=abort\n");
          printf("(disk might be WRITE_PROTECTED?)\n");
          if (!index("\r\n ", (char) Cnecin()))
              longjmp(start, USER_ABORT);
          else
              goto redo;
         }
     } else {
         i++;
     }
    }
}

/*
 * copy_disk: where the copy is really done
 */

copy_disk()
{
    if((read_source || verify_source)
        && (drives_content[d1] != SOURCE_DISK))
```

```
        prompt_for_disk(SOURCE_DISK, d1, s1);
    if (read_source) {      /* source */
     read_disk(d1, s1); /* read */
     next_line();
    }
    if (verify_source && (!multipass || first_pass)) {
     verify_disk(d1, s1);
     next_line();
    }

    setjmp(start_dest);     /* destination */
    if(zero_disk) do_zero_disk();              /* blank image */
    if(!override) randomize_boot();            /* change serial */
    if((format || Verify || write_dest || verify))
     prompt_for_disk(DEST_DISK, d2, s2);
    if (format && (!multipass || first_pass)) {         /* format */
     format_disk(d2, sides - 1, s2);
     next_line();
    }
    if (Verify && (!multipass || first_pass)) {         /* Verify */
     verify_disk(d2, s2);
     next_line();
    }
    if (write_dest) {       /* write */
     write_disk(d2, s2);
     next_line();
    }
    if (verify && (!multipass || last_pass)) {     /* verify */
     verify_disk(d2, s2);
     next_line();
    }
}

/*
 * return 1 if skip, 0 if no key, longjmps to start if abort
 */

abort_on_key(track,side)
int             track,side;
{
    if (Cconis()) {
     if ((int) Cnecin() == 0x1b)
         longjmp(start, USER_ABORT);
     printf("\n\007KEY PRESSED! CR/SP = continue, ");
     printf("i=interactive, %s other abort:\n",
            (track == -1 ? "" : " S = skip,"));
     switch ((int) Cnecin()) {
     case ' ':
     case '\n':
     case '\r':
         return 0;
     case 'i':
     case 'I':
```

```
        interactive = 1;
        return 0;
    case 's':
    case 'S':
        if (track != -1) {
         printf("Ok, skipping track %d, side %d (sects #%d - #%d)\n",
                track + 1, side +1,
                ((track * sides) + side)* st,
                ((track * sides) + side + 1) * st);
         return 1;
        }
    }
    longjmp(start, USER_ABORT);
   }
}

int
retry(s, t, side, restart)
char           *s;
int             t, side;
long           *restart;
{
    printf("\n\007** %s ERROR track %d ,side %d (s. #%d - #%d) ",
        s, t + 1, side + 1,
        (t * sides + side) * st,
        (t * sides + side + 1) * st);
    printf("CR/SP retry, S skip, %sother abort",
        (restart ? "BS=OTHER disk " : ""));
    fflush(stdout);
    switch ((int) Cnecin()) {
    case ' ':
    case '\r':
    case '\n':
     return 1;
    case 's':
    case 'S':
     printf("\n OK, skipping track %d, side %d\n", t + 1, side + 1);
     return 0;
    case '\010':
     if (restart)
         longjmp(restart, 1);
    default:
     longjmp(start, USER_ABORT);
    }
}

/*
 * sector_read
 * reads one track into ptr, reading sector by sector
 * returns: 0=ok, 1=bad, 2= skip.
 */

int
```

```
sector_read(ptr, foo, drive, sect1, track, side, n)
char           *ptr, *foo;
int             drive, sect1, track, side, n;
{
    int             i, result = 0, pass;
    int             pass_ok[4];

    for (pass = 0; pass < 4; pass++) {
     pass_ok[pass] = 1;
     for (i = 0; i < n; i++) {
         if (abort_on_key(track,i))
          return 2;
         switch (one_sector_read(
           (pass ? buffer_data + (pass - 1) * 5120 : ptr) + (i * bps),
                       foo, drive, i + sect1, track, side)) {
         case 1:
          if (!result)
              printf("\nTrack %d, side %d: sectors ",
              track + 1, side + 1);
          result = 1;
          printf("%d (#%d) ", i + 1, (track * sides + side) * st + i);
          fflush(stdout);
          bfill(ptr + i * bps, '\0', (long) bps); /* fills bad sector with */
                               /* 0's */
         case 2:
          pass_ok[pass] = 0;
         }
     }
     if (result) {
         printf("are bad!\n");
         fflush(stdout);
         return result;
     }
    }
    if (pass_ok[0])
     return 0;     /* if any pass is good, use it */
    {
     int             i;

     for (i = 1; i < 4; i++) {
         if (pass_ok[i]) {
          bcopy(buffer_data + 512 * (i - 1), ptr, 512 * n);
          return 0;
         }
     }
    }
    if (check_random(track, side, n, ptr, buffer_data)) {
     return 1;     /* random read */
    } else {
     printf("\nTrack %d, side %d, (sects #%d - #%d) read sect by sect\n"
            ,track + 1, side + 1,
            (track * sides + side) * st,
            (track * sides + side + 1) * st);
```

```
     return 1;     /* no but don't trust! */
    }
}

/*
 * check_random:
 * check tracks buffer to see if equal, return 1 if error
 */

int
check_random(track, side, n, buf0, buf1)
int             track, side, n;
char           *buf0, *buf1;
{
    int             sect, i, bad = 0;
    char           *p0, *p1, *p2, *p3;

    for (sect = 1; sect <= n; sect++) {
     p0 = buf0 + (sect - 1) * 512;
     p1 = buf1 + (sect - 1) * 512;
     p2 = buf1 + (sect - 1) * 512 + 5120;
     p3 = buf1 + (sect - 1) * 512 + 10240;
     for (i = 0; i < 512; i++, p0++, p1++, p2++, p3++) {
         if ((*p0 != *p1) || (*p0 != *p2) || (*p0 != *p3)) {
          if (!bad) {
              printf("\n\007DIFFERENT READS ");
              printf("on track %d, side %d, sects ",
                  track + 1, side + 1);
              bad = 1;
          }
          printf("%d (#%d) ", sect,
          (track * sides + side) * st + sect - 1);
          break;
         }
     }
    }
    if(bad) {
        printf("\n");
     fflush(stdout);
    }
    return bad;
}


/*
 * one_sector_read:
 * reads a sector and the 2 outside it
 * return 0 if ok, 1 if bad, 2 if forced to copy sector alone
 */

char           *private_buffer[512 * 3];

int
```

```
one_sector_read(buf, foo, drive, sect, track, side)
char           *buf, *foo;
int             drive, sect, track, side;
{
    if (Floprd(private_buffer + (sect == 1 ? 512 : 0), foo, drive,
            (sect == 1 ? 1 : sect - 1), track, side,
            ((sect == 1) || (sect == st) ? 2 : 3))) {
     if (Floprd(private_buffer + 512, foo, drive, sect, track, side, 1)){
         return 1;
     } else {
         bcopy(private_buffer + 512, buf, 512);
         return 2;
     }
    }
    bcopy(private_buffer + 512, buf, 512);
    return 0;
}

next_line()
{
    Crawio((int) '\r');
    Crawio((int) '\n');
}

bcopy(from, to, count)
register char  *from, *to;
int             count;
{
    register char  *last = from + count;

    while (from != last)
     *to++ = *from++;
}

bfill(from, octet, count)
register char  *from, octet;
long            count;
{
    register char  *last = from + count;

    while (from != last)
     *from++ = octet;
}

/*
 * is_drive_B
 * return 1 if drive B is there
 */

int
is_drive_B()
{
    long            usermode = Super(0L);
```

```
    int             result = *(int *) 0x4a6;   /* _nflops */

    Super(usermode);
    return (result - 1);
}

/*
 * retry_track
 * retries plain track
 */

int
retry_track(track, side)
int             track, side;
{
    printf("\n\007** Track-read error track %d ,side %d \n", track + 1, side + 1);
    printf("CR/SP retry, ESC aborts, other go to sector_read: ");
    fflush(stdout);
    switch ((char) Cnecin()) {
    case ' ':
    case '\r':
     return 1;
    case 0x1b:
     longjmp(start, USER_ABORT);
    default:
     return 0;
    }
}

verify_disk(drive, side)
int             drive, side;
{
    char           *ptr, *map;
    int             t, n;

    if(multipass)
        map = full_track_map;
    else
     map = track_map;
    ptr = buffer_data;
    for (t = 0; t < TRACK_MAP_SIZE; t++) {
     if (map[t]) {
         for (n = side; n < (side + sides); n++) {
          if (Floprd(ptr, 0L, drive, 1, t, n, st))
              while (retry("Verify", t, n, start_dest) &&
                  Floprd(ptr, 0L, drive, 1, t, n, st));
          abort_on_key(t,n);
         }
         Crawio((int) 'v');
     }
    }
}
```

```
write_disk(drive, side)
int             drive, side;
{
    char           *ptr;
    int             t, n, last_track;

    ptr = buf;
    if(zero_disk){          /* formatting is a special case */
     last_track = 2;
    }else{
     last_track = TRACK_MAP_SIZE;
    }
    for (t = 0; t < last_track; t++) {
     if (track_map[t]) {
         for (n = side; n < (side + sides); n++) {
          if (Flopwr(ptr, 0L, drive, 1, t, n, st))
              while (retry("Write", t, n, start_dest) &&
                  Flopwr(ptr, 0L, drive, 1, t, n, st));
          ptr += (long) bps *st;

          abort_on_key(-1,n);
         }
         Crawio((int) '.');
     }
    }
}

read_disk(drive, side)
int             drive, side;
{
    char           *ptr = buf, c;
    int             t, n;
    int             full_track_retry;

    /* we need at least room for 1 track! */
    if(buffer_size < (sides * bps * st * 1L)) longjmp(start, NO_MEMORY);
    for (t = 0; t < TRACK_MAP_SIZE; t++) {
     if (track_map[t]) {
         c = ',';
         for (n = side; n < (side + sides); n++) {
          if (Floprd(ptr, 0L, drive, 1, t, n, st)) {
              c = ';';
              full_track_retry = 1;
              while(1) {
               abort_on_key(-1,n);
               if (!Floprd(ptr, 0L, drive, 1, t, n, st))
                   goto good_retry;
               full_track_retry++;
               if (full_track_retry >= TRACK_RETRY) {
                   if (interactive && retry_track(t, n)) {
                       full_track_retry = 1;
                   }else{
                       break;
```

```
                    }
                }
               }
               /* 4 retries done */
               if (1 == sector_read(ptr, 0L, drive, 1, t, n, st)) {
                if (interactive) {
                    while (retry("Read", t, n, 0L) &&
                    (1 == sector_read(ptr, 0L, drive, 1, t, n, st)));
                }
               }
           }
      good_retry:
           ptr += (long) (bps * st);
           abort_on_key(-1,n);
          }
          Crawio((int) c);
      }
     }
}

input_track_map()
{
    char            text[10];
    int             i, strack, etrack;

    for (i = 0; i < TRACK_MAP_SIZE; i++)
     track_map[i] = '\0';
    while ((printf("Enter start track (CR = done): "),
         gets(text), (int) text[0])) {
     strack = atoi(text) - 1;
     printf("Enter end track (CR = %d): ", strack + 1);
     gets(text);
     if (!text[0])
         etrack = strack;
     else
         etrack = atoi(text) - 1;
     if ((etrack > TRACK_MAP_SIZE) ||
         (strack > etrack) ||
         (strack < 0)) {
         printf("Sorry, not a valid entry: %d...%d\n",
             strack + 1, etrack + 1);
     } else {
         for (i = strack; i <= etrack; i++)
          track_map[i] = '\001';
         printf("Ok, range %d...%d will be done\n",
             strack + 1, etrack + 1);
     }
    }
}

/*
 * prompt_for_disk:
 * ask user to insert disk if we think it is not in drive
```

```
 */

prompt_for_disk(disk, drive, side)
int disk, drive,side;
{
    if(drives_content[drive] != disk){
     printf("\nInsert %s disk in drive %c (side %d),",
         (disk == SOURCE_DISK ? "SOURCE" : "DESTINATION"),
         (char)(drive + 'A'), side + 1);
     printf("CR/SP = ok, other = abort\n");
     if (!index("\r\n ", (char) Cnecin())){
         if(disk == SOURCE_DISK)
          one_valid_copy_done = 0;
         longjmp(start, USER_ABORT);
     }else{
         drives_content[drive] = disk;
     }
    }
}

/*
 * update_checksum:
 * compute checksum of boot buffer and set it (last int of 256 ints)
 */

update_checksum(block, is_boot_non_executable)
unsigned block[];
unsigned is_boot_non_executable;
{
    int i; unsigned sum = 0;
    for(i=0; i<255; i++)
     sum += block[i];
    block[255] = (unsigned) 0x1234 - sum + is_boot_non_executable;
}

unsigned
is_non_executable(block)
unsigned block[];
{
    int i; unsigned sum = 0;
    for(i=0; i<=255; i++)
     sum += block[i];
    return (sum == (unsigned) 0x1234 ? (unsigned) 0 : (unsigned) 1);
}

@\Rogue\Monster\
else
  echo "shar: Will not over write FLIP.C"
fi
echo "Finished archive 1 of 2"
# to concatenate archives, remove anything after this line
exit 0
```

From -1934850369541309917
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,1d8e5a36491cbeb3,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-24 16:08:46 PST
Path: utzoo!utgpu!water!watmath!clyde!att!osu-cis!tut.cis.ohio-state.edu!bloom-beacon!mit-eddie!ll-xn!ames!pasteur!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i035: flip -- disk copier, part02/02
Keywords: Megamax
Message-ID: <3413@saturn.ucsc.edu>
Date: 24 May 88 23:08:46 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1273
Approved: koreth@ssyx.ucsc.edu

Submitted-by: uunet!mcvax!mirsa.inria.fr!colas (Colas Nahaboo)
Posting-number: Volume 1, Issue 35
Archive-name: flip/part02

```
#--------------------------Cut Here--------------------------
#! /bin/sh
# This is a shell archive.  Remove anything before the "#! /bin/sh" line,
# then unpack it by saving it in a file and typing "sh file."
#
# Wrapped by Steven Grimm (koreth) at ssyx on Tue May 24 15:54:23 1988
#
# unpacks with default permissions
#
# Contents : MAKEFILE MAN.C README TWST.C
#
if `test ! -s MAKEFILE`
then
echo "x - MAKEFILE"
cat > MAKEFILE << '@\Rogue\Monster\'
VERSION=26b

flip_$(VERSION).tos:   flip.o twst.o man.o
     mmlink -o flip_$(VERSION).tos flip.o twst.o man.o

imp: flip.o twst.o man.o
     mmimp flip.o
     mmimp twst.o
     mmimp man.o
     mmlink -o flip_$(VERSION).tos flip.o twst.o man.o

@\Rogue\Monster\
else
  echo "shar: Will not over write MAKEFILE"
fi
if `test ! -s MAN.C`
```

```
then
echo "x - MAN.C"
cat > MAN.C << '@\Rogue\Monster\'
/* FLIP HISTORY & MANUAL */

/* HISTORY:
 *=========
 *   2.6b tells when copies an executable boot to detect virii
 *   2.6a bug de 'r' repaired
 *        No checksum done if no previous checksum
 *        option 's' for single instead of double (double=default)
 * v 2.6  27 fevrier 88
 *   2.5c re-designing of command parsing
 *    formats disk ('f' command)
 *    randomization of serial # at each copy
 *   2.5b t can override disk
 *    ESC aborts without confirm
 *   2.5a no ESC to another disk, BS!
 *    aborts on key even in format
 *    man in another C file
 * v 2.5  released
 *  2.4.5 write-protect detected
 *  2.4.3 option v to verify source only
 *    multipass copies possible
 *  2.4.2 all sects# outputted on any kind of error.
 *        twister modified to accept starttrack parameter
 *    track_map to know what to copy
 *    t option to specify track
 *    r takes options fvV
 *  2.4.1 i for interactive: default no stop
 *    V verify after format (case-sensitive options)
 *    bug of verify reading into main buffer repaired
 * v 2.4: Jan 15 88
 *  2.3.2 If esc on insert abort
 *    track_read first
 *    no more skip opt
 *    # of sect printed
 *  2.3.1 Drive B presence is really tested
 * v 2.3: Dec 28 87
 *  2.2.3 comparisons were not done!!! -- repaired
 *    copy by 3 sectors
 *        stops if copy sector only by 1
 *  2.2.2 bug de a1a1 repare
 *  2.2.1 no verify by default
 *    r for re-copy
 * v 2.2: Dec 19 87
 *  2.1.1 prints buffer size on start
 *    sector_read done 4 times with comparison!
 *    RETURN repeats last command
 * v 2.1: Nov 28 87
 *  2.0.1 retrys in writes and verify work now.
 *    verify there is a drive B before copying
 * v 2.0: Nov 22 87
```

```
 *  1.6.3 verify is another pass
 *  1.6.2 format does not verify anymore
 *     format outputs a char
 *  1.6.1 uses TWISTER's code mmtwst.c to format
 * v 1.6: Nov 11 87
 *  1.5.4 abort on ESC
 *        option "2" for 2-sided.
 *  1.5.3 abort on key with confirm
 *  1.5.2 track, sectors begin at 1 instead of 0
 *        retrys on writes retry format-write-verify loop
 *     on write errors, ESC to go to another disk
 *  1.5.1 port to megamax
 *     uses all memory
 * v 1.5: more checking of entry syntax.
 *        on-line help
 *        fills bad sectors with 0s before writing
 *     more sensible memory allocation
 *     s option for auto skip
 * v 1.4: one automatic retry without prompting. If skips, copies good
 *     sectors anyway.
 * v 1.3: when error on a track, asks for retrys, which are done sector
 *        by sector.
 * v 1.2: when double sided, copy the two sides
 * v 1.1: whole side in memory for one drive use.
 * v 1.0: original one in mark williams. quick and really dirty.
 */

/* ON_LINE_MANUAL:
 *================
 */

char           *help_text[] = {
"",
"                  FLIP 2.6 manual",
"                   Colas Nahaboo",
"",
"    flip - disk copier with error-recovery, on either sides.",
"           with FAST formatting as TWISTER or DCFORMAT",
"",
"SYNTAX",
"  the commands are given as: (items in [] are optional)",
"",
"      <source-drive>[<dest-drive>][fvVtios9]                 : copy disks",
"      f[<drive>][fvVtis9]                                    : format disk",
"      v[<drive>][tios9]                                      : verify disk",
"      r[<drive>][fvVi]                              : re-write disk in memory",
"      q                                                      : quits",
"      ?                                                      : help",
"",
"  A drive is specified by the letter \"a\" or \"b\", followed optionally by",
"  \"1\" or \"2\" to indicate the side. (default drive is \"a\", side is \"1\")",
"",
"OPTIONS",
```

```
"     f - to format destination side",
"     V - verify disk by re-reading track just after being formatted.",
"     v - verify copy by re-reading track just after being written",
"     t - to specify what track(s) to copy (used with \"z\" to give the",
"           number of tracks of the new disk, default 80)",
"     i - interactive: ask user in case of error (default don't)",
"     o - override: do not look at the source disk boot sector to determine",
"           its type, take 10 sectors, 1 sided, 80 tracks, unless told",
"           otherwise by options \"9\", \"d\" or \"t\"",
"     9 - disk is 9 sectors per track (default 10). Used with \"z\" or \"o\"",
"     s - disk is single-sided (default double). Used with \"z\" or \"o\"",
"",
"EXAMPLES",
"  ab2fv - to copy a non-protected SS disk in drive A on the unformatted",
"          back of a SS disk in drive B, verifying the copy.",
"  afv   - (or fv) to copy ANY disk with format and verify, in drive A.",
"  rfv   - to re-write last copy into drive A",
"  ffv   - to format a double-sided disk in drive A",
"  fs    - to erase all files of a previously formatted single sided disk",
"",
"OPERATION",
"  During copy, flip prints for each track, a:",
"  ,    for reading track without problem",
"  ;    for reading track successfully after 4 retrys sector-by-sector",
"  -    for formatting a track (in FAST format)",
"  v    for verifying a track",
"",
"  You can pause during the copy at any time by pressing a key, but pressing",
"  ESC will abort. ^C always quits.",
"",
"  Without i option, on read error, it retries 4 times to read the track,",
"  then reads sector by sector. With option i, it prompts you for what to do:",
"  (the track and sector number given start at 1, but sector number at 0)",
"    - RETURN or SPACE retries. You'd better try to extract, then re-insert",
"      the disk to try to re-center it. (or try on another drive later)",
"    - \"s\" skips the bad sectors, going on with the rest of the copy.",
"      the bad sectors will be filled with 0s on the copy",
"      (but all that could be read is kept, of course!)",
"    - if it is writing the copy, you can press BACKSPACE if you want to try",
"      to write to another disk.",
"    - any other key aborts the copy.",
"  Experience shows that you can recover most of your damaged disks this way.",
"  If it cannot read the track, it reads it sector by sector, but tells you so",
"  because due to a bug in the TOS, a sector can be read incorrectly without",
"  giving an error. So FLIP does 4 reads of the same sector and compares",
"  them to be sure, but even if reads are equals, it warns you because the",
"  risk subsists, due to the bug in old TOS roms...",
"",
"DESCRIPTION",
"  The program copies TOS disks as a whole, and only non-protected",
"  ones. It can consider the back side of a single-sided disk as another disk",
"  so, it is very useful to use the back sides of protected disks, or",
"  single sided disks, such as MAGIC! ones. (with a two-side drive, of",
```

```
"  course!)",
"",
"  But it also reads disks with care, being able to read CORRECTLY most disks",
"  causing a read error to copiers or the GEM desktop. Thus, if you get a",
"  read error on a disk, copy it with FLIP!. Moreover, if it cannot read a",
"  sector, it tells you so! (Unlike the HELP program)",
"",
"  Version 2.6 can format and changes randomly the serial # of the disk",
"  when copying, (without the override option). It copies also executable",
"  boot sectors, but tells you so (in 2.6b) so that you can check for a VIRUS.",
"  it gives you the values at $3A offset to identify the virus",
"  Known virus values are: 41,FA,FF                                        ",
"                                                                          ",
"  When it reads a disk, it first tries to read the boot sector in order",
"  to know the size (tracks and sectors per track) of the disk. If it",
"  can't make sense out of it, it uses 80 tracks by 9 sector, unless you",
"  used the \"o\" option",
"  Thus you don't have to tell him the type of the disk you want to copy.",
"  It will recognize double sided disks automatically, but you can only",
"  copy double sided disks, without flipping their faces!",
"",
"  So, NEVER COPY A DISK WICH GIVES READ ERRORS WITH A BIT-COPIER (procopy)",
"  use FLIP instead!",
"",
"  From version 2.0 on, FLIP format in FAST format, all disks accesses on",
"  this disk will be twice faster compared to a normal disk!. It ALWAYS ",
"  formats disks in 10 sectors, even if only to store 9 sectors.",
"  Be aware that I didn't test the validy of the formatting code on the",
"  new roms, so make some tests if you have them.",
"",
"COPYRIGHT",
"  This program is PUBLIC DOMAIN, and should be given with its sources",
"  Feel free to improve it!, but please mention my name.",
"  I would be very glad if you send me back any improved version, such as",
"  a GEM port",
"  CREDIT: It uses code from TWISTER from Dave SMALL & Dan MOORE.",
"",
"BUGS",
"  Not under GEM - yet!",
"  Cannot copy IBM boot sectors (use DCFORMAT)",
"",
"AUTHOR",
"  Colas NAHABOO",
"  383 ch du clos d'Embertrand",
"  06250 MOUGINS",
"  FRANCE.",
"  Personal phone: 93 75 68 29, give me a call...",
"  Electronic mail address:    colas@mirsa.inria.fr",
0L
};

#include "osbind.h"
#include "stdio.h"
```

```
more(text)
char          **text;
{
    char          **p = text, c;
    int             line = 0;

    while (*p) {
     printf("%s\n", *p++);
     line++;
     if (line > 22) {
         printf("%cp more (SP, CR, BS or ESC) %cq", 27, 27);
         fflush(stdout);
         while (!Cconis());
         c = Cnecin();
         printf("%cM", 27);
         fflush(stdout);
         switch (c) {
         case 27:
         case 'q':
          printf("\n");
          return;
          break;
         case ' ':
          line = 0;
          break;
         case 8:
          line = 0;
          p -= 48;
          if (p < text)
              p = text;
          printf("%cp...Backing 1 page...%cq\n", 27, 27);
          break;
         }
     }
    }
    printf("\n");
}

@\Rogue\Monster\
else
  echo "shar: Will not over write MAN.C"
fi
if `test ! -s README`
then
echo "x - README"
cat > README << '@\Rogue\Monster\'
Flip is a small copier that I wrote for my personal use to:

- use the back of SS disks

- try to save data from damaged disks (getting frustrated with the abort
  message of DCFORMAT)
```

Since I and my friends are using it all the time, it might be of use to some
netlanders. As it was developed for my own use, it is not under GEM.
It is in no way as great a program as is GULAM or UNITERM

It is completly public domain, files are: (MEGAMAX C v1.1)

binaries:
     flip_26b.tos   the executable

sources:
     makefile
     flip.c   main code
     man.c   manual & history
     twst.c   formatter in in-line assembly, in fact TWISTER code
               slightly modified (MEGAMAX v1.1)
```
@\Rogue\Monster\
else
  echo "shar: Will not over write README"
fi
if `test ! -s TWST.C`
then
echo "x - TWST.C"
cat > TWST.C << '@\Rogue\Monster\'
/*
 *       Megamax C inline ASM version of Twister.
 *       Converted from AS68 format  dlm  12/14/86.
 *       AS68 version by David Small before that date.
 *       Used for the MegAMin HD backup program.
 *
 */

/* patched by colas for use with FLIP:
 * add params:
 *   flipped: 0= no, 1=use back as front (must be SS then)
 *   endtrack: 80 for normal disks
 *   starttrack: 0 normally
 * outputs a '-' for each formatted cylinder (see  PUTCHAR comments)
 * no more verify
 */

/* routine that are callable from C are: */
extern twister();
extern confirm_abort();
extern printf();

#include "osbind.h"
#include "stdio.h"

/* global variables */
int thedisk, dblsided, sectoroffset;   /* input parameters */
int flipped, endtrack, starttrack, keypress;     /* colas params */
long buffer;
```

```
int badflag;                        /* bad sector error flag */

/* local variables */
static int theside, thetrack, twistsectorno, sectorno, retrycnt,
          cdev, ctrack, csect, cside, ccount, A7t, interlv,
          virgin, def_error, curr_err;
static long cdma, edma, tmpdma, saveA2;
static int saveA0, saveA1, saveD0;
/* now the defines (AS68 equ's) */

/*  tunable values */
#define retries   2         /* default # of retries -1 */
#define midretry  1         /* "middle" retry, when to reseek  */
#define timeout   0x40000   /* short timeout, motor already on */
#define ltimeout  0x60000   /* long timeout, to startup motor
     */

/*  Floppy state variables in DSB: RAM usage */
#define recal    0xff00     /* recalibrate flag
     */
#define dcurtrack 0         /* current track number
     */
#define dseekrt   dcurtrack+2 /* seek rate     */
#define dsbsiz    dseekrt+2 /* size of a DSB   */

/* hardware equates */
/* DMA chip        */
#define diskctl    0xffff8604    /* disk controller data access */
#define fifo       0xffff8606    /* DMA mode control / status
      */
#define dmahigh    0xffff8609    /* DMA base high
      */
#define dmamid     0xffff860b    /* DMA base mid
      */
#define dmalow     0xffff860d    /* DMA base low
      */

/* 1170 select values  */
#define cmdreg     0x80          /* select command register
     */
#define trkreg     0x82          /* select track register
     */
#define secreg     0x84          /* select sector register
     */
#define datareg    0x86          /* select data register
     */

/* GI sound chip (drive a/b select lines and side select) */
#define giselect   0xffff8800    /* (W) sound chip register select */
#define giread     0xffff8800    /* (R) sound chip read data
     */
#define giwrite    0xffff8802    /* (W) sound chip write data
```

```
    */
#define giporta    0x0e          /* GI register # for i/o port A */

/* 68901 mfp chip */
#define mfp        0xfffffa00    /* mfp base             */
#define gpip       mfp+1         /* general purpose i/o
    */

/* misc defines */
#define seekrate 3
#define dsb0    0xa06
#define dsb1    0xa0a
#define flock   0x43e

char *abort_message = "\n\007KEY PRESSED! CR/SP = continue, other abort:\n";

asm {         /* start of inline ASM */

/*
 *   All the following is munged code from AS68 (that's why the
 *   weird format).  The AS68 code was munged from the BIOS
 *   source.  And most of this is from the FDC article in Start
 *   issue number 3.
 *
 *   Anyway, we apologize for the mess, but heh it works.
 *
 *   Oh yes.  The labels are usually on nops, not empty lines.  Thats
 *   because AS68 occasionally pukes on labels on empty lines.  And
 *   I didn't take the time to remove them.  (Megamax doesn't care.)
 *
 */

;*****************************************************************
; Twister: Faster formatter (zipformat) hack.  MM version
        *
; Copyright (c) 1986, 1987 START Magazine (as usual)
        *
; Written by David Y. Small & Dan Moore                    *
;*****************************************************************
;
;              Dedication:
;
;     For Bill The Cat: because he's twisted, too.
;
;*****************************************************************
;
; code begins.
;
;  PASS ME THIS STUFF DAN! Then SHAVE! For God's sakes! What
; are you trying to do, look like Fidel Castro?
;
; And get SOME BEER in here okay? Sheesh!
;
```

```
; input integers (if coming from C):
;
; dblsided: 0000 or ffff
; thedisk: 0 = a, 1 = b
;************************************************************


;*******************************************************
; Frammatter
;*******************************************************
twister:  move.l   A2, saveA2(A4)     ; everything else saved by the Supexec

        clr.w      badflag(A4)        ; assume no errors
;
;*** Init vars for formatting loop
;
     move.w D0, saveD0(A4);
     move.w starttrack(A4), D0   ;start at starttrack
     move.w D0, thetrack(A4)
     move.w saveD0(A4), D0
;WAS    move.w  #0,thetrack(A4)   ; start at track 0

        move.w  #1,twistsectorno(A4) ; twister value

     move.w A0, saveA0(A4)
     move.w A1, saveA1(A4)
trackloop: nop                ; comes here for subseq tracks
     move.w saveA0(A4), A0
     move.w saveA1(A4), A1

;***** FRONT SIDE ***********
;**** Setup stack: .. based on read sec command, tweaked for frammat

        move.w  #0xe5e5,-(A7)    ; virgin data -- e5's work okay
        move.l  #0x87654321,-(A7) ; magic # to make format work / fmt only
        move.w  #1,-(A7)       ; sector interleave factor / fmt only
        move.w  flipped(A4),-(A7)       ; ** side.w (relevant) / side: FRONT
        move.w  thetrack(A4),-(A7)  ; ** track.w (relevant) / same
        move.w  #10,-(A7)      ; sector.w (irrelevant) / sectors per track
        move.w  thedisk(A4),-(A7)      ; ** device.w  (relevant) / same
        move.l  #0,-(A7)       ; dummy.l (irreleveant) / same
        move.l  buffer(A4),-(A7) ; ** buffer address.l (relevant) / same
        jsr     _flopfmt
        adda.l  #24,A7         ; Fix stack.
        cmpi.w  #0xffff, badflag(A4)   ; aborts on error
        beq     gemexit
;****
; Jazz twistsectorno(A4). Look at what it ended up as; set it to
; the next track twist depending on that.
;
; It will end up as "+1" from the last sector formatted, because
; the formatter bumps it up.
;
```

```
        cmpi.w  #1,twistsectorno(A4)      ; 0-1 transition
        beq     fgoto1
        cmpi.w  #9,twistsectorno(A4)      ; 1-2 transition
        beq     fgoto2
        cmpi.w  #7,twistsectorno(A4)      ; 2-3 transition
        beq     fgoto3
        cmpi.w  #5,twistsectorno(A4)      ; 3-4 transition
        beq     fgoto4
        cmpi.w  #3,twistsectorno(A4)      ; 4-0 transition
        beq     fgoto0

        dc.b  0x4a
        dc.b  0xfc                ; oops, die (illegal instruction)
;*****
; Really class coding, eh? Oh,well, it's easy and it works, which
; twelve other fancy ways of doing this *don't*.
;
fgoto1: move.w  #0x9,twistsectorno(A4)         ; 9,a,1,2,3,4,5,6,7,8
        bra     fkeepformatting
fgoto2: move.w  #0x7,twistsectorno(A4)         ; 7,8,9,a,1,2,3,4,5,6
        bra     fkeepformatting
fgoto3: move.w  #0x5,twistsectorno(A4)         ; 5,6,7,8,9,a,1,2,3,4
        bra     fkeepformatting
fgoto4: move.w  #0x3,twistsectorno(A4)         ; 3,4,5,6,7,8,9,a,1,2
        bra     fkeepformatting
fgoto0: move.w  #0x1,twistsectorno(A4)         ; 1,2,3,4,5,6,7,8,9,a
        bra     fkeepformatting
        nop
;*****
fkeepformatting: nop
;*****
; now, do a verify on those sectors.
;**** Setup stack: .. based on read sec command, tweaked for frammat
;
;     tst.w    sectoroffset(A4)  ; no verify if 11 to 20 sector
;     ; numbering
;     bne  nover1
;
;     move.w  #0xa,-(A7)        ; count.w 2
;     move.w  flipped(A4),-(A7)    ; ** side.w (relevant) / side: FRONT 4
;     move.w  thetrack(A4),-(A7)  ; ** track.w (relevant) / same 6
;     move.w  #0x1,-(A7)        ; sector.w (relevant) / starting sector 8
;     move.w  thedisk(A4),-(A7)   ; ** device.w  (relevant) / same 0xa
;     move.l  #0,-(A7)    ; dummy.l (irreleveant) / same 0xe
;     move.l  buffer(A4),-(A7) ; ** buffer address.l (relevant) / same 0x12
;
;     move.w  #19,-(A7)   ; do BIOS _flopver 0x14
;     trap     #14
;     adda.l  #20, A7         ; fix stack.
;
;     tst.l    D0
;     bne  badsecs
;
```

```
;****

;***** BACK SIDE ******     (How kinky Dave.  What _are_ you talking about?)

nover1:  tst.w   dblsided(A4)
         beq     nobackside

;**** Setup stack: .. based on read sec command, tweaked for frammat

        move.w  #0xe5e5,-(A7)    ; virgin data -- e5's work okay
; and with a virgin.  My, you really are weird Dave.
        move.l  #0x87654321,-(A7) ; magic # to make format work / fmt only
        move.w  #1,-(A7)       ; sector interleave factor / fmt only
        move.w  #1,-(A7)       ; ** side.w (relevant) / side: BACK
        move.w  thetrack(A4),-(A7)  ; ** track.w (relevant) / same
        move.w  #10,-(A7)      ; sector.w (irrelevant) / sectors per track
        move.w  thedisk(A4),-(A7)      ; ** device.w  (relevant) / same
        move.l  #0,-(A7)       ; dummy.l (irreleveant) / same
        move.l  buffer(A4),-(A7) ; ** buffer address.l (relevant) / same
        jsr     _flopfmt
        adda.l  #24,A7         ; Fix stack.
        cmpi.w  #0xffff, badflag(A4)   ; aborts on error
        beq     gemexit
;*****
;
; Jazz twistsectorno(A4). Look at what it ended up as; set it to
; the next track twist depending on that.
;
; It will end up as "+1" from the last sector formatted, because
; the formatter bumps it up.
;
        cmpi.w  #1,twistsectorno(A4)      ; 0-1 transition
        beq     goto1
        cmpi.w  #9,twistsectorno(A4)      ; 1-2 transition
        beq     goto2
        cmpi.w  #7,twistsectorno(A4)      ; 2-3 transition
        beq     goto3
        cmpi.w  #5,twistsectorno(A4)      ; 3-4 transition
        beq     goto4
        cmpi.w  #3,twistsectorno(A4)      ; 4-0 transition
        beq     goto0

        dc.b    0x4a
        dc.b    0xfc               ; oops, die

;
goto1:  move.w  #0x9,twistsectorno(A4)         ; 9,a,1,2,3,4,5,6,7,8
        bra     keepformatting
goto2:  move.w  #0x7,twistsectorno(A4)         ; 7,8,9,a,1,2,3,4,5,6
        bra     keepformatting
goto3:  move.w  #0x5,twistsectorno(A4)         ; 5,6,7,8,9,a,1,2,3,4
        bra     keepformatting
goto4:  move.w  #0x3,twistsectorno(A4)         ; 3,4,5,6,7,8,9,a,1,2
```

```
        bra     keepformatting
goto0:  move.w  #0x1,twistsectorno(A4)         ; 1,2,3,4,5,6,7,8,9,a
        bra     keepformatting
        nop
;
keepformatting: nop           ; entry after twisting.
;
;
;     ;**** Setup stack: .. based on read sec command, tweaked for frammat
;
;     tst.w     sectoroffset(A4) ; no verify for strange sector numbers
;     bne  nobackside
;
;     move.w  #0xa,-(A7)        ; count.w 2
;     move.w  #1,-(A7)    ; ** side.w (relevant) / side: BACK 4
;     move.w  thetrack(A4),-(A7)  ; ** track.w (relevant) / same 6
;     move.w  #0x1,-(A7)        ; sector.w (relevant) / starting sector # 8
;     move.w  thedisk(A4),-(A7)   ; ** device.w  (relevant) / same 0xa
;     move.l  #0,-(A7)    ; dummy.l (irreleveant) / same 0xe
;     move.l  buffer(A4),-(A7) ; ** buffer address.l (relevant) / same 0x12
;
;     move.w  #19,-(A7)   ; do BIOS _flopver 0x14 (why not)
;     trap      #14
;     adda.l  #20, A7          ; fix stack
;
;     tst.l     D0
;     bne  badsecs        ; wham
;
;****
nobackside: nop      ; entry if not DS  (well at least this isn't kinky Dave)

; add 1 to thetrack. If endtrack, quit.
        addi.w   #1, thetrack(A4)
     move.w A0, saveA0(A4)
     move.w A1, saveA1(A4)

     move.w D0, saveD0(A4)       /* PUTCHAR */
     move.w #0x2D, -(A7)         /* outputs a '-' */
     move.w #6, -(A7)
     trap #1
     adda.l #4, A7
     move.w #0xB, -(A7)      /* CNONIS */
     trap #1
     adda.l #2, A7
     cmpi.w  #0,D0
         bne key_pressed              /* aborts on keypress */
resume:  move.w saveD0(A4), D0       /* PUTCHAR end */

     move.w endtrack(A4), A0
     move.w thetrack(A4), A1
     cmpa A0, A1
        bne     trackloop      ; "do another"
```

```
     move.w saveA0(A4), A0
     move.w saveA1(A4), A1
        bra    gemexit        ; Continue, etc, and so forth

;*********
key_pressed:
     MOVE.w   #8,-(A7)
     trap #1
     ADDa.L   #2,A7
     CMPI.w   #27,D0
     beq  key_abort
     move.l  abort_message(A4),-(A7)
     move.w  #9,-(A7)
     trap #1
     ADDa.L   #6,A7
     MOVE.w   #8,-(A7)
     trap #1
     ADDa.L   #2,A7
     CMPI.w   #32,D0
     beq  resume
     CMPI.w   #10,D0
     beq  resume
     CMPI.w   #13,D0
     beq  resume
key_abort:    MOVE.w   #1,keypress(A4)
     move.w  saveD0(A4), D0
     bra gemexit

;*********
badsecs: move.w #0xffff, badflag(A4)  ; bad sectors found

;**************
;
; End. Exit to calling routine.
;
gemexit: movea.l saveA2(A4), A2  ; only reg not saved by Supexec
         rts

;*****************************************************
;
;-----------------------------------------------------------------------
;       130-ST / 520-ST
;       Floppy Disk Driver
;       (C)1985 Atari Corp.
;       From the FDC article (Start issue 3); shortened to just
;       the formatter routine and the low level I/O.

;
;-----------------------------------------------------------------------
;
;*************************************************************
;
; _flopfmt - format a track
```

```
; Passed (on the stack):
;       0x1a(A7) initial sector data
;       0x16(A7) magic number
;       0x14(A7) interleave
;       0x12(A7) side
;       0x10(A7) track
;        0xe(A7) A7t(A4)
;        0xc(A7) drive
;        0x8(A7) pointer to state block
;        0x4(A7) dma address
;        0x0(A7) [return]
;
; Returns:     EQ: track successfully written.  Zero.W-terminated list of
;              bad sectors left in buffer (they might /all/ be bad.)
;
;              NE: could not write track (write-protected, drive
;              failure, or something catastrophic happened).
;-
_flopfmt:
        cmpi.l   #0x87654321, 0x16(A7)        ; check for magic# on stack
        bne     flopfail          ; no magic, so we just saved the world
;       bsr     change            ; check for disk flip
;       moveq   #e_error,D0       ; set default error number
        bsr     floplock              ; lock floppies, setup parms
        bsr     select                ; select drive and side
        move.w  0xe(A7),A7t(A4)       ; save sectors-per-track
        move.w  0x14(A7),interlv(A4)      ; save interleave factor
        move.w  0x1a(A7),virgin(A4)       ; save initial sector data

;--- put drive into "changed" mode
;       moveq   #m_changed,D0         ; D0 = "CHANGED"
;       bsr     setdmode              ; set media change mode

;--- seek to track (hard seek):
;debug* move.l  #0xff00ff00,0xf8030
        bsr     hseek                 ; hard seek to 'ctrack'
;debug* move.l  #0x00000000,0xf8030
        bne     flopfail              ; (return error on seek failure)
        move.w  ctrack(A4),dcurtrack(A1)    ; record current track#

;--- format track, then verify it:
;*      move.w  #e_error,curr_err(A4)(A5)   ; vanilla error mode
;debug* move.l  #0xff00ff00,0xfA030
        bsr     fmtrack               ; format track
;debug* move.l  #0x0,0xfA030

        bne     flopfail              ; (return error on seek failure)
        bra     flopok


;****************************
;+
; fmtrack - format a track. Tweaked for skewed interleave.
```

```
;
; Passed:      variables setup by _flopfmt
; Returns:     NE on failure, EQ on success
; Uses:        almost everything
; Called-by:   _flopfmt
;
;-
fmtrack:
;*      move.w  #e_write,def_error(A4)    ; set default error number
        move.w  #1,D3        ; start with sector 1, first pass
;**
        movea.l  cdma(A4),A2          ; A2 =  prototyping area
; Lay down beginning of track

        move.w  #60-1,D1              ; 60 x 0x4e (track leadin)
        move.b  #0x4e,D0
        bsr     wmult
; Repeat 10 times: sector data.
        move.w  #1,sectorno(A4)           ; how many secs written
; Note that twistsectorno is initialized out of this routine.
;--- address mark
secloop: nop
;
ot1:    move.w  #12-1,D1              ; 12 x 0x00
        clr.b   D0
        bsr     wmult
        move.w  #3-1,D1               ; 3 x 0xf5
        move.b  #0xf5,D0
        bsr     wmult
        move.b  #0xfe,(A2)+            ; 0xfe -- address mark intro
        move.b  ctrack+1(A4),(A2)+        ; track# - low half of word
        move.b  cside+1(A4),(A2)+             ; side# - low half of word

;* #1: just use a plain sector number
;*      move.b  D4,(A2)+              ; sector#

;* #2: use a twisted sector number
;*      move.b  twistsectorno+1(A4),(A2)+   ; new: sector #.

        move.w  twistsectorno(A4),D0      ; fetch    1-10 sector #
;
; Note that meg-a-minute backup uses sectors # 11-20 to force GEM
; not to use MM disks.
;
        add.w   sectoroffset(A4),D0   ; add possible shift to 11-20 sector #
        move.b  D0,(A2)+              ; plug it into sector table

; Add 1 to sector #. If it is b, wrap it to 1.
        addi.w  #1,twistsectorno(A4)      ; real sector # being plopped
        cmpi.w  #0xb,twistsectorno(A4)
        bne     notb
        move.w  #1,twistsectorno(A4)
;
```

```
notb:   nop
        move.b  #0x02,(A2)+           ; sector size (512)
        move.b  #0xf7,(A2)+           ; write checksum

;--- gap between AM and data:
        move.w  #22-1,D1             ; 22 x 0x4e
        move.b  #0x4e,D0
        bsr     wmult
        move.w  #12-1,D1             ; 12 x 0x00
        clr.b   D0
        bsr     wmult
        move.w  #3-1,D1              ; 3 x 0xf5
        move.b  #0xf5,D0
        bsr     wmult

;--- data block:
        move.b  #0xfb,(A2)+           ; 0xfb -- data intro
        move.w  #256-1,D1        ; 256 x virgin.w (initial sector data)
ot2:    move.b  virgin(A4),(A2)+         ; copy high byte
        move.b  virgin+1(A4),(A2)+       ; copy low byte
        dbf     D1,ot2               ; fill 512 bytes
        move.b  #0xf7,(A2)+           ; 0xf7 -- write checksum
        move.w  #40-1,D1             ; 40 x 0x4e
        move.b  #0x4e,D0
        bsr     wmult
;
; Next sector, or, end-of-track. If we hit sector 11, time to quit.
;
        addi.w   #1,sectorno(A4)
        cmpi.w  #11,sectorno(A4)
        bne     secloop        ; loop again
;
; Okay, 10 sectors laid down.
;
;--- end-of-track
        move.w  #1400,D1         ; 1401 x 0x4e -- end of track trailer
        move.b  #0x4e,D0
        bsr     wmult

;--- setup to write the track:
        move.b  cdma+3(A4),dmalow     ; load dma pointer
        move.b  cdma+2(A4),dmamid
        move.b  cdma+1(A4),dmahigh
        move.w  #0x190,(A6)           ; toggle R/W flag and
        move.w  #0x090,(A6)           ; select sector-count register
        move.w  #0x190,(A6)
        move.w  #0x1f,D7              ; (absurd sector count)
        bsr     wdiskctl
        move.w  #0x180,(A6)           ; select 1770 cmd register
        move.w  #0xf0,D7              ; write format_track command
;debug* move.l  #0xff00ff00,0xfc030
        bsr     wdiskctl
        move.l  #timeout,D7          ; D7 = timeout value
```

```
;--- wait for 1770 to complete:
otw1:   btst.b  #5,gpip              ; is 1770 done?
        beq     otw2                 ; (yes)
        subq.l  #1,D7                ; if(--D7) continue;
        bne     otw1
;debug* move.l  #0xffffffff,0xfc030

        bsr     reset1770            ; timed out -- reset 1770
oterr:  moveq   #1,D7                ; return NE (error status)
        rts

;--- see if the write-track won:
otw2:   nop
;debug* move.l  #0x0,0xfc030

        move.w  #0x190,(A6)           ; check DMA status bit
        move.w  (A6),D0
        btst    #0,D0             ; if its zero, there was a DMA error
        beq     oterr                ; (so return NE)
        move.w  #0x180,(A6)           ; get 1770 status
        bsr     rdiskctl
;**     bsr     err_bits             ; set 1770 error bits
        and.b   #0x44,D0  ; check for writeProtect & lostData
        rts                  ; return NE on 1770 error

;------ write 'D1+1' copies of D0.B into A2, A2+1, ...
wmult:  move.b  D0,(A2)+             ; record byte in proto buffer
        dbf     D1,wmult             ; (do it again)
        rts
;
;*************************************************************
;
; floplock - lock floppies and setup floppy parameters
;
; Passed (on the stack):
;       0x18(A7) - count.W (sector count)
;       0x16(A7) - side.W (side#)
;       0x14(A7) - track.W (track#)
;       0x12(A7) - sect.W (sector#)
;       0x10(A7) - dev.W (device#)
;        0xc(A7) - obsolete.L
;        8(A7) - dma.L (dma pointer)
;        4(A7) - ret1.L (caller's return address)
;        0(A7) - ret.L (floplock's return address)
;
; It is not coincidence that this matches the read and write input list;
; they call us to get stuff off stack and plug into parameters. Then,
; routine "select" actually pops this stuff into hardware.
;
; Passed:       D0.W = default error number
;
; Also, we helpfully point A6 at the DMA chip, and A1 at the DSB.
```

```
;-
floplock:
        lea   regsave, A0
        movem.l D3-D7/A3-A6, (A0)         ; save C registers
        lea    fifo,A6               ; A6 -> fifo
; Start setting up param block..
        move.w  D0,def_error(A4)         ; set default error number
        move.w  D0,curr_err(A4)          ; set current error number
; Kick VBL off floppies..
        move.w  #1,flock              ; tell vbl not to touch floppies
; Stuff off stack:
        move.l  8(A7),cdma(A4)           ; cdma -> /even/ DMA address
        move.w  0x10(A7),cdev(A4)       ; save device# (0 .. 1)
        move.w  0x12(A7),csect(A4)   ; save sector# (1 .. 9, usually)
        move.w  0x14(A7),ctrack(A4)  ; save track# (0 .. 39 .. 79 ..)
        move.w  0x16(A7),cside(A4)   ; save side# (0 .. 1)
        move.w  0x18(A7),ccount(A4)  ; save sector count (1..A)
;--- pick a DSB: Point A1 at it.
        lea    dsb0,A1        ; pick dsb 0 (drive A)
        tst.w  cdev(A4)
        beq    flock2
        lea    dsb1,A1        ; pick dsb 1 (drive B)
;--- compute ending DMA address from count parameter: Plug into edma.
; This is used in multisector transfers in read-multiple-sector,
; but we don't use it here.
flock2: moveq  #0,D7
        move.w  ccount(A4),D7            ; edma = cdma + (ccount * 512)
        lsl.w  #8,D7
        lsl.w  #1,D7                  ; do a 9 shift..
        movea.l  cdma(A4),A0
        adda.l   D7,A0
        move.l  A0,edma(A4)              ; save in edma
;--- recalibrate drive, if it needs it. This only happens when the DSB
; says that this drive has never awoken before, and needs an initial
; recal to match its current track # with the DSB track #.
        tst.w  dcurtrack(A1)         ; if (curtrack < 0) recalibrate()
        bpl    flockr
; RECAL needed. Show flag about it..
        move.l  #0xf0f0f0f0,0xfe000
        move.l  #0xf0f0f0f0,0xfe004

        bsr    select                ; select drive & side
        clr.w  dcurtrack(A1)         ; we're optimistic -- assume winnage
; Restore.
        bsr    restore               ; attempt restore
        beq    flockr                ; (it won)
        moveq  #10,D7                ; attempt seek to track 10
        bsr    hseek1
        bne    flock1                ; (failed)
        bsr    restore               ; attempt restore again
        beq    flockr                ; (it won)
;
flock1:  move.w  #recal, dcurtrack(A1)      ; complete failure (what can we do?)
```

```
        movem.l regsave(PC), D3-D7/A3-A6
flockr: rts
;
;*********************************************************************
;+
; flopfail - unlock floppies and return error.
; Common way for read and write to return.
;
; Note: Returns via unlok1.
;-
flopfail:
        move.w #0xffff, badflag(A4)           ; bad sectors found
        move.l  #0xffffffff,D0         ; aargh, error
        bra   unlok1

;*************************************************
;+
; flopok - unlock floppies and return success status. Also a common
; way for r/w to return.
;
;-
flopok: clr.l  D0                     ; return 0 (success)
; Entry point from flopfail..
unlok1: move.l  D0,-(A7)              ; (save return value)
; I believe this code returns the FDC's status to a type-1 status, where
; the write protect switch is available for VBL to look at.
        move.w  #datareg,(A6)         ; force WP to real-time mode
; Sets FDC's current track register to track we are on right now.
        move.w  dcurtrack(A1),D7      ; dest-track = current track
        bsr    wdiskctl
; Does it with a "noop seek" (source=dest). Only forces FDC to type 1 status.
        move.w  #0x10,D6               ; cmd = seek w/o verify
        bsr    flopcmds               ; do it
;
unlok2: move.l  (A7)+,D0              ; restore return value
        movem.l  regsave(PC), D3-D7/A3-A6
; clear floppy lock of vblank..
        clr.w  flock        ; allow vblank .. unlock floppies
; wave byebye at the pretty camera, jenny...
        rts
;*********************** Seek routines ********************
;+
; hseek  - seek to 'ctrack(A4)' without verify
; hseek1 - seek to 'D7' without verify
; hseek2 - seek to 'D7' without verify, keep current error number
;
; Returns:     NE on seek failure ("cannot happen"?)
;              EQ if seek wins
;
; Uses:        D7, D6, ...
; Exits to:    flopcmds
; Called-by:   _flopfmt, _flopini
```

```
;
;-
hseek:  move.w  ctrack(A4),D7            ; dest track = 'ctrack'
hseek1: nop
hseek2: move.w  #datareg,(A6)        ; write destination track# to data reg
        bsr     wdiskctl             ; write D7 to FDC data register
; seek command:
        move.w  #0x10,D6              ; execute "seek" command
;                                    * Note: no spinup time.
        bra     flopcmds             ; (without verify...)
;
;*********************************************
;+
; reseek - home head, then reseek track
; Returns:     EQ/NE on success/failure
; Falls-into:  go2track
;
;-
reseek:
        bsr     restore              ; restore head
        bne     go2trr               ; (punt if home fails)

        clr.w   dcurtrack(A1)        ; current track = 0
        move.w  #trkreg,(A6)         ; set "current track" reg on 1770
        clr.w   D7
        bsr     wdiskctl             ; write a 00 to trk register.

        move.w  #datareg,(A6)        ; seek out to track five
        move.w  #5,D7
        bsr     wdiskctl             ; dest track = 5
        move.w  #0x10,D6
        bsr     flopcmds             ; seek
;                                    * Note: no spinup time.
        bne     go2trr               ; return error on seek failure
        move.w  #5,dcurtrack(A1)     ; set current track#

;*********************************************+
; go2track - seek proper track
; Passed:      Current floppy parameters (ctrack, et al.).
; Returns:     EQ/NE on success/failure
; Calls:       flopcmds
; Called by: read sector, for instance. Lots of places.
;-
go2track:
        move.w  #datareg,(A6)        ; set destination track# in
        move.w  ctrack(A4),D7            ;  1770's data register
        bsr     wdiskctl             ; (write track#)
        moveq   #0x14,D6              ; execute 1770 "seek_with_verify"
        bsr     flopcmds             ; (include seek-rate bits)
        bne     go2trr               ; return error on seek failure
        move.w  ctrack(A4),dcurtrack(A1)    ; update current track number
        and.b   #0x18,D7              ; check for RNF, CRC_error, lost_data
go2trr: rts                          ; return EQ/NE on succes/failure
```

```
;************************************************
;+
; restore - home head
; Passed:      nothing
; Returns:     EQ/NE on success/failure
;-
restore:
        clr.w   D6                      ; 0x00 = 1770 "restore" command
        bsr     flopcmds                ; do restore
        bne     res_r                   ; punt on timeout
        btst    #2,D7                   ; test TRK00 bit
        eori.w  #0x04, CCR              ; flip Z bit (return NE if bit is zero)
        bne     res_r                   ; punt if didn't win
        clr.w   dcurtrack(A1)           ; set current track#
res_r:  rts

;************************************************
; Special floppy cmd just for seeking:
;+
; flopcmds - floppy command (or-in seek speed bits from database)
; Passed:      D6.w = 1770 command
; Sets-up:     seek bits (bits 0 and 1) in D6.w
; Falls-into:  flopcmd
; Returns:     EQ/NE on success/failure
;
; I get the impression this is only used for seeking. I am right.
;-
flopcmds:
        move.w  dseekrt(A1),D0          ; get floppy's seek rate bits
        andi.b  #3,D0                   ; OR into command
        or.b    D0,D6
; Fall in...
;*****************************************************+
; flopcmd - execute any ol' 1770 command (with timeout)
; Passed:      D6.w = 1770 command
;
; Returns:     EQ/NE on success/failure
;              D7 = 1770 status bits
;
; Note: does motor spinup, if cmd in D6 specs it (I would *hope*!)
;-
flopcmd:
        move.l  #timeout,D7             ; setup timeout count (assume short)
        move.w  #cmdreg,(A6)            ; select 1770 command register
        bsr     rdiskctl                ; read it to clobber READY status
        btst    #7,D0                   ; is motor on?
        bne     flopcm                  ; (yes, keep short timeout)
        move.l  #ltimeout,D7            ; extra timeout for motor startup
flopcm: bsr     wdiskct6                ; write command (in D6)

flopc1: subq.l  #1,D7                   ; timeout?
        beq     flopcto                 ; (yes, reset and return failure)
```

```
        btst.b  #5,gpip             ; 1770 completion?
        bne     flopc1              ; (not yet, so wait some more)
        bsr     rdiskct7            ; return EQ + 1770 status in D7
        clr.w   D6
        rts
;****
flopcto:
; We timed out.. a gruesome death indeed.
; Whap controller back to life.  (Hit him harder Dave, he likes it that way)
        bsr     reset1770           ; bash controller
        moveq   #1,D6               ; and return NE
        rts

;*****************************************************
;+
; reset1770 - reset disk controller after a catastrophe
; Passed:       nothing
; Returns:      nothing
; Uses:         D7
;-
reset1770:
        move.w  #cmdreg,(A6)        ; execute 1770 "reset" command
        move.w  #0xD0,D7             ; force interrupt
        bsr     wdiskctl
        move.w  #15,D7              ; wait for 1770 to stop convulsing
r1770:  dbf     D7,r1770            ; (short delay loop)
        bsr     rdiskct7            ; return 1770 status in D7
        rts

;*****************************************************
;+
; select - setup drive select, 1770 and DMA registers
; Passed:       cside(A4), cdev(A4)
; Returns:      appropriate drive and side selected
;
; Called: All over the place.
;
;-
select: move.w  #0,0x9c0             ; floppies NOT deselected
;                                   * lets VBL shut them off...
        move.w  cdev(A4),D0             ; get device number
        addq.b  #1,D0               ; add and shift to get select bits
        lsl.b   #1,D0               ; into bits 1 and 2
        or.w    cside(A4),D0            ; or-in side number (bit 0)
        eori.b   #7,D0               ; negate bits for funky hardware select
        andi.b   #7,D0               ; strip anything else out there
        bsr     setporta            ; do drive select
; Have to restore 1770's track register from table..in dsb.
        move.w  #trkreg,(A6)        ; setup 1770 track register
        move.w  dcurtrack(A1),D7    ; from current track number
        bsr     wdiskctl
        clr.b   tmpdma(A4)              ; zero bits 24..32 of target DMA addr
```

```
; Setup R/W parameters on 1770. Used by
; r/w sector, among others. This is where the sector register gets set.
;
select1:
        move.w  #secreg,(A6)          ; setup requested sector_number from
        move.w  csect(A4),D7              ;       caller's parameters
        bsr     wdiskctl
        move.b  cdma+3(A4),dmalow             ; setup DMA chip's DMA pointer
        move.b  cdma+2(A4),dmamid
        move.b  cdma+1(A4),dmahigh
        rts

;************************************************
;+
; setporta - set floppy select bits in PORT A on the sound chip
; Passed:       D0.b (low three bits)
; Returns:      D1 = value written to port A
;               D2 = old value read from port A
; Uses:         D1
;-
setporta:
        move    SR,-(A7)              ; save our IPL
        ori     #0x0700,SR             ; start critical section
        move.b  #giporta,giselect         ; select port on GI chip
        move.b  giread,D1             ; get current bits
        move.b  D1,D2                 ; save old bits for caller
        andi.b   #0xff-7,D1             ; strip low three bits there
        or.b    D0,D1                 ; or-in our new bits
        move.b  D1,giwrite            ; and write 'em back out there
        move    (A7)+,SR              ; restore IPL to terminate CS, return
        rts

;************************************************
;+
; Primitives to read/write 1770 controller chip (DISKCTL register).
;
; The 1770 can't keep up with full-tilt CPU accesses, so
; we have to surround reads and writes with delay loops.
; This is not really as slow as it sounds.
;
wdiskct6:                             ; write D6 to diskctl
        bsr     rwdelay               ;       delay
        move.w  D6,diskctl            ;       write it
        bra     rwdelay               ;       delay and return

wdiskctl:                             ; write D7 to diskctl
        bsr     rwdelay               ;       delay
        move.w  D7,diskctl            ;       write it
        bra     rwdelay               ;       delay and return

rdiskct7:                             ; read diskctl into D7
        bsr     rwdelay               ;       delay
        move.w  diskctl,D7            ;       read it
```

```
        bra     rwdelay                 ;       delay and return

rdiskctl:                               ; read diskctl into D0
        bsr     rwdelay                 ;       delay
        move.w  diskctl,D0              ;       read it
; And here's the delay loop:
rwdelay:
        move    SR,-(A7)                ; save flags
        move.w  D7,-(A7)                ; save counter register
        move.w  #0x20,D7                ; 0x20 seems about right...
rwdly1: dbf     D7,rwdly1               ; busy-loop: give 1770 time to settle
        move.w  (A7)+,D7                ; restore register, flags, and return
        move    (A7)+,SR
        rts

;-------------------------------------
regsave:  dc.l      1       ; PC relative data area
          dc.l      2
          dc.l      3
          dc.l      4
          dc.l      5
          dc.l      6
          dc.l      7
          dc.l      8
          dc.l      9


}       /* MY GOD!  It's DONE!  */

@\Rogue\Monster\
else
  echo "shar: Will not over write TWST.C"
fi
echo "Finished archive 2 of 2"
# to concatenate archives, remove anything after this line
exit 0


From -98677817681547749
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,208529f61cff12d2,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-27 11:39:31 PST
Path: utzoo!attcan!uunet!lll-winken!lll-tis!ames!umd5!purdue!decwrl!ucbvax!agate!
saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i036: rzsz -- UNIX side for zmdm file transfer program  part01/05
Keywords: UNIX
Message-ID: <3483@saturn.ucsc.edu>
Date: 27 May 88 18:39:31 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1160
```

```
Approved: koreth@ssyx.ucsc.edu

Submitted-by: bammi@mandrill.ces.cwru.edu (Jwahar R. Bammi)
Posting-number: Volume 1, Issue 36
Archive-name: rzsz/part01

[This is the UNIX end to the zmdm file transfer program in comp.binaries.
 It provides for batch transfers, among other things.  You'll probably
 only want this if you plan to use zmdm. -sg]

#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
# This is part 1 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
#
#    Run the following text with /bin/sh to create:
#      Makefile
#      README
#      expandar.c
#      gz
#      minirb.1
#      minirb.c
#      rbsb.c
#      rz.1
#      rz.c
#      sz.1
#      sz.c
#      zm.c
#      zmodem.h
#
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
sed 's/^X//' << 'SHAR_EOF' > Makefile &&
X#
X# Please note:
X#   Define the symbol `CASE' only if you want the CWRU
X#   enhancements AND you are running on a BSD 4.N system.
X#
X
X#all: rz sz minirb
Xall: rz sz
X
X#CC = gcc
X#CFLAGS=-O -DCRCTABLE -DV7 -DCASE
XCC = cc
XCFLAGS=-O -DCRCTABLE -DV7 -DCASE -v -mc68020
X
X# define BIN appropriately
XBIN=/usr/local/bin
X
```

```
Xrz: rz.o
X    $(CC) -o rz rz.o -s -x -n
X#   $(CC) -o rz rz.o
X
Xsz: sz.o expandargs.o
X    $(CC) -o sz sz.o expandargs.o -s -x -n
X#   $(CC) -o sz sz.o expandargs.o
X
X#minirb:
X#   $(CC) $(CFLAGS) -o minirb minirb.c -s -x -n
X
Xrz.o: rz.c rbsb.c zm.c zmodem.h
X
Xsz.o: sz.c rbsb.c zm.c zmodem.h
X
X# define BIN appropriately
X# You may need to change `ln -s' to the appropriate
X# link command if you are not running BSD4.N
X#
X#install: sz rz minirb
Xinstall: sz rz
X    rm -f $(BIN)/rb $(BIN)/sb $(BIN)/rz $(BIN)/sz $(BIN)/minirb
X#   mv rz sz minirb $(BIN)
X    mv rz sz $(BIN)
X    ln -s $(BIN)/rz $(BIN)/rb
X    ln -s $(BIN)/sz $(BIN)/sb
X
Xclean:
X    rm -f *.o report
SHAR_EOF
chmod 0600 Makefile || echo "restore of Makefile fails"
sed 's/^X//' << 'SHAR_EOF' > README &&
XNOTE::
X    Users of zmdm on the Atari ST's please see note at end of file.
X-------------------------------------------------------------------------------
XCompile instructions are located in the comments at the beginning of the rz.c
Xans sz.c files.  The main compiler flags are -DV7 for V7 and BSD systems and
X-DSVR2 for sVr2.  Other #ifdefs in the files may be activated to deal with
Xvarious types of kernel brain damage if you have a mutant system.
X
XNaturally, rz ans sz work best with comm programs that seamlessly support ZMODEM
Xcommand and file AutoDownload (PowerCOm, Pro-YAM and ZCOMM).  Ths DOS "DSZ"
Xsub-program allows ZMODEM file transfers with older programs, but it must be
Xcalled manually.  DSZ and the ZMODEM protocol description are on TeleGodzilla
Xand other bulletin boards.  They may be obtained via UUCP from site "omen", for
Xwhich a direct UUCP login and telephone number are provided below.  You may uucp
X~uucp/FILES first to determine the pathnames for the "goodies".
X
XChuck Forsberg WA7KGX Author of Pro-YAM communications Tools for PCDOS and Unix
X...!tektronix!reed!omen!caf  Omen Technology Inc "The High Reliability Software"
X  17505-V Northwest Sauvie Island Road Portland OR 97231  Voice: 503-621-3406
XTeleGodzilla BBS: 621-3746 2400/1200  CIS:70007,2304  Genie:CAF  Source:TCE022
X  omen Any ACU 1200 1-503-621-3746 se:--se: link ord: Giznoid in:--in: uucp
```

```
X  omen!/usr/spool/uucppublic/FILES lists all uucp-able files, updated hourly
X
X--------------------------------------------------------------------------
X
X         Case Western Reserve U 'enhancements':
X    (feed back about these enhancements to bammi@mandrill.cwru.edu)
X
XThis distribution contains some enhancements to the code as distributed
Xby Chuck Forsberg. These enhancements will only work if the proprocessor
Xsymbol 'CASE' if defined. PLEASE NOTE THAT THESE ENHANCEMENTS HAVE ONLY
XBEEN TESTED ON Unix (Vax/Sun) BSD4.X SYSTEMS. They should work on
Xother flavors of BSD, but WILL NOT WORK ON ANY OTHER FLAVORS OF UNIX
X(ie. when you compile make sure that the preprocessor symbol `CASE' is
XNOT defined, if you are not using BSD Unix).
X
XThe 'enhancements':
X
X1) When sending files with SZ, if one of the arguments is the name
Xof a directory, then all the files in that directory, and all the
Xsub-directories thereof will be sent. If you want the complete
Xpath names to be sent also, use the -f option of SZ. Also, to give
Xbetter control the  -P name switch was added. -P name prunes the
Xsubdirectory named, so that it is not included. The arguement to
X-P can be the name of a sub-directory, of the name of a file. If it
Xis the name of a sub-directory, then that branch of the file hierarcy
Xis 'pruned' and is not sent. If it the name of a file, then that
Xparticular file is not sent. For example, if you were backing up the
Xfile system of your unix system (hypothetical of course :-) to your ST,
Xand did'nt want to backup /tmp amd /etc, then you would issue the
Xfollowing command to your unix system:
X
X    sz -f -P /tmp -P /etc /
X
X    (and come back the next day!!)
X
X2) When receiving files with RZ, if the sender is sending full
Xpath names (-f option to SZ), then all directories that do not exist,
Xwill be automatically created by RZ.
X
X3) sz now allows you to send long blocks at low baud rates (<= 2400 baud).
X    try sz -l 1024 files ...
XEarlier it insisted on 256 byte blocks max at 2400 baud or less.
X
X    Any comments and/or suggestions about these enhancements should
Xbe directed to
Xusenet:      {decvax,sun}!cwruecmp!bammi jwahar r. bammi
Xarpa/csnet:  bammi@mandrill.cwru.edu
XcompuServe:  71515,155
SHAR_EOF
chmod 0600 README || echo "restore of README fails"
sed 's/^X//' << 'SHAR_EOF' > expandar.c &&
X#ifdef CASE
X
```

```
X    /* CWRU specific extention(hack :-)
X     *   Examine each argument given to main,
X     *   If it is a directory, then expand it
X     *   to all its component files, recursively
X     *   till you bottom out.
X     *   If it is not a directory, then just pass it on.
X     *
X     *   Inputs: argc, argv
X     *   Outputs: nargc, nargv (calls MAIN(nargc, nargv))
X     *   To test: compile with -DCASE -DTEST
X     *         run with a directory as an arg
X     *   Hacker: JRB
X     *   Requirements: BSD 4.2/3 specific routines used
X     *              only tested on Vax BSD 4.3
X     *
X     *   Added -P name prune option at the suggestion of dietz@zhmti
X     *   Multiple -P's may be given on the command line.
X     *   -P name may be given anywhere on the command line. -P name
X     *   will prune the subdirectories named as arguement to -P.
X     *   ie: when the program is decending the file hierarchy it
X     *   will not visit the pruned branches.
X     *   NOTE that the option is -P and not -p
X     *         (-p is a valid sz option).
X     *   NOTE that the arguement given to -P can be the name
X     *   of a file or a directory. In case it is a file,
X     *   that file is skipped.
X     *
X     */
X
X/*
X * Expand argc, so that the called MAIN(nargc,nargv) receives only
X * filenames, in nargv[][]
X *
X *  WARNING: Contains BSD specific code, tested on BSD4.3 ONLY.
X *
X *  Jwahar Bammi
X *  usenet: {decvax,cbatt,cbosgd,sun}!cwruecmp!bammi
X *  csnet:       bammi@case
X *  arpa:        bammi%case@csnet-relay
X *  compuServe:  71515,155
X *
X */
X#include <stdio.h>
X
X#define OK    0
X#define FALSE     0
X#define Realloc    realloc
X
Xtypedef struct _prunelist {
X    char *name;             /* name of subdirectory to prune */
X    struct _prunelist *next;    /* ptr to next */
X} PRUNELIST;
X
```

```
Xstatic char *ProgName;
Xstatic PRUNELIST *PruneList = (PRUNELIST *)NULL; /* Head of PruneList */
X
X
Xmain(argc, argv)
Xint argc;
Xchar **argv;
X{
X    register int status;
X    int nargc = 0;
X    char **nargv = (char **)NULL;
X    extern int MAIN();
X    extern char **CopyToNargv();
X    extern PRUNELIST *AddPrune();
X
X    ProgName = *argv;
X
X    /* copy argv[0] blindly */
X    if((nargv = CopyToNargv(*argv, nargc, nargv)) == (char **)NULL)
X        return(~OK);
X
X    nargc++;
X
X    while((--argc) > 0)
X    {
X         argv++;
X         if(**argv == '-')
X         {
X              /* copy any options except -P */
X              if( (*argv)[1] == 'P')
X              {
X                   if((--argc) <= 0)
X                   {
X                        fprintf(stderr,"no argument given to -P\n");
X                        return(~OK);
X                   }
X                   if((PruneList = AddPrune(PruneList,*++argv))
X                      == (PRUNELIST *)NULL)
X                       return(~OK);
X              }
X              else
X              {
X                   if((nargv = CopyToNargv(*argv, nargc, nargv))
X                      == (char **)NULL)
X                       return(~OK);
X                   else
X                       nargc++;
X              }
X
X         }
X         else
X         {
X              /* If its not on the PruneList then */
```

```
X             if(!OnPruneList(PruneList, *argv))
X             {
X                  /* if it is a directory, push it */
X                  if(isdir(*argv))
X                  {
X                       if((status = PushDir(*argv)) != OK)
X                       {
X                            return(status);
X                       }
X                  }
X                  else
X                  {
X                      /* it is NOT a directory, copy to nargv */
X                       if((nargv = CopyToNargv(*argv, nargc, nargv))
X                          == (char **)NULL)
X                           return(~OK);
X                       else
X                           nargc++;
X                  }
X             }
X         }
X    } /* while */
X
X    /* process pushed directories if any */
X    if((status = ProcessDirs(&nargc, &nargv)) != OK)
X        return(status);
X    /* else Free the Stack and call MAIN */
X    FreeStack();
X    return (MAIN(nargc, nargv));
X}
X
X#include <sys/types.h>
X#include <sys/stat.h>
X
X/*
X * is name a directory ?
X *
X */
Xint isdir(name)
Xchar *name;
X{
X    struct stat st;
X
X    if(stat(name, &st) != 0)
X        return (FALSE);    /* assume its not a directory */
X
X    /* NOTE: the expression below eliminates Sockets.
X       (st.st_mode & S_IFMT)& S_IFDIR does not!
X     */
X    return( ( (st.st_mode & S_IFMT) == S_IFDIR) );
X}
X
X/*
```

```
X * Expand nargv by an element and copy a String into the new element
X *
X */
Xchar **CopyToNargv(string, nargc, nargv)
Xchar *string;
Xint nargc;
Xchar **nargv;
X{
X    extern char *malloc(), *Realloc(), *strcpy();
X    extern int strlen();
X
X    /* expand nargv by 1 element */
X    if(nargv == (char **)NULL)
X        /* do it with malloc for the first one */
X        nargv = (char **)malloc(sizeof(char **));
X    else
X        /* do it with Realloc for others */
X        nargv = (char **)Realloc(nargv, (nargc+1)*sizeof(char **));
X
X    if(nargv == (char **)NULL)
X    {
X         /* failed to get memory */
X         perror(ProgName);
X         return ((char **)NULL);
X    }
X
X    /* Get mem for string */
X    if(( nargv[nargc] = malloc(strlen(string)+1)) == (char *)NULL)
X    {
X         /* failed to get memory */
X         perror(ProgName);
X         return ((char **)NULL);
X    }
X
X    /* copy string into nargv[nargc] */
X    (void)strcpy( nargv[nargc], string);
X    return(nargv);
X}
X
Xstatic char **Stack = (char **)NULL; /* directory stack */
Xstatic char StackSize = 0;      /* Size of current Stack */
Xstatic int    Top   = -1;
X#define STACK_EMPTY   (Top < 0)
X#define CHUNKSIZE 16
X
X/*
X * Grow the Stack by one chunk of CHUNKSIZE elements
X *
X */
Xchar **ExpandStack(Stack)
Xchar **Stack;
X{
X    extern char *malloc(), *Realloc();
```

```
X
X    /* Grow Stack */
X    if(Stack == (char **)NULL)
X        /* with malloc */
X        Stack = (char **)malloc(CHUNKSIZE * sizeof(char **));
X    else
X        /* with Realloc */
X        Stack = (char **)Realloc(Stack, (StackSize+CHUNKSIZE) *
X                                 sizeof(char **));
X
X    if(Stack == (char **)NULL)
X    {
X         /* outa mem */
X         perror(ProgName);
X         return((char **)NULL);
X    }
X    StackSize += CHUNKSIZE;
X    return(Stack);
X}
X
X/*
X * Free the Stack
X *
X */
XFreeStack()
X{
X    if(StackSize > 0)
X         (void)free(Stack);
X}
X
X
X/*
X * Push a directory name on Stack
X *
X */
Xint PushDir(name)
Xchar *name;
X{
X    register int status;
X    extern char *malloc(), *strcpy();
X    extern int strlen();
X    extern char **ExpandStack();
X
X    ++Top;
X    if(Top >= StackSize)
X    {
X         if((Stack = ExpandStack(Stack)) == (char **)NULL)
X             return(~OK);
X    }
X
X    if((Stack[Top] = malloc(strlen(name)+1)) == (char *)NULL)
X    {
X         /* outa mem */
```

```
X         perror(ProgName);
X         return(~OK);
X    }
X    (void)strcpy(Stack[Top], name);
X    return(OK);
X}
X
X/*
X * Pop a directory name from the stack
X *
X */
Xchar *PopDir()
X{
X    register char *r;
X    extern char **ShrinkStack();
X
X    if(STACK_EMPTY)
X        return ((char *)NULL);
X
X    r = Stack[Top];
X    Top--;
X    return(r);
X}
X
Xstatic int BadStatus = FALSE;
X#define BADSTATUS (BadStatus != FALSE)
X
X#include <sys/dir.h>
X/*
X * Process directories on the Stack, by adding all the
X * files in a directory to nargv.
X */
Xint ProcessDirs(nargc, nargv)
Xint *nargc;
Xchar ***nargv;
X{
X    register char *name;
X    register DIR *dirp;
X    register struct direct *dp;
X    register int status;
X    char path[MAXNAMLEN+1];
X    extern DIR *opendir();
X    extern struct direct *readdir();
X    extern char **CopyToNargv();
X    extern char *PopDir();
X
X    if(BADSTATUS)
X        return(~OK);
X
X    if(STACK_EMPTY)
X        /* Nothing more to do */
X        return(OK);
X
```

```
X    /* Pop a directory from Stack and process */
X    if((name = PopDir()) == (char *)NULL)
X    {
X         /* Oh Oh */
X         fprintf(stderr,"Internal Error (BUG), PopDir returns NULL\n");
X         BadStatus = (~FALSE);
X         return(~OK);
X    }
X
X    /* Open the directory */
X    if((dirp = opendir(name)) == (DIR *)NULL)
X    {
X         /* trouble opening directory */
X         perror(name);
X         /* set BADSTATUS and return */
X         BadStatus = (~FALSE);
X         return(~OK);
X    }
X
X    /* for each entry in the directory, if it is a file
X       add to nargv. If it is a directory, Push it onto
X       the directory stack.
X     */
X    for (dp = readdir(dirp); dp != (struct direct *)NULL;
X         dp = readdir(dirp))
X    {
X         if( (!((strcmp(dp->d_name,".") == 0) ||
X               (strcmp(dp->d_name,"..") == 0))) &&
X            (dp->d_ino != 0 ))  /* not a free entry */
X         {
X              strcpy(path, name);
X              strcat(path,"/");
X              strcat(path,dp->d_name);
X
X              /* If this path is on the PruneList skip */
X              if(OnPruneList(PruneList, path))
X                  continue;
X
X              if(!isdir(path))
X              {
X                   /* not a dir -- add this to nargv */
X                   if((*nargv = CopyToNargv(path, *nargc, *nargv))
X                      == (char **)NULL)
X                   {
X                        closedir(dirp);
X                        BadStatus = (~FALSE);
X                        return(~OK);
X                   }
X                   else
X                   {
X                        *nargc += 1;
X                   }
X
```

```
X             }
X             else
X             {
X                  /* Note: At this point we could have
X                     simply recursed on ProcessDirs()
X                     totally eliminating the directory Stack.
X                     This was not done as it is possible
X                     to run out of dirp's on deep
X                     file hierarchies. Yet another (ir)rational
X                     design decision.
X                   */
X
X                  /* Push This directory */
X                  if((status = PushDir(path)) != OK)
X                  {
X                       closedir(dirp);
X                       BadStatus = (~FALSE);
X                       return(status);
X                  }
X             }
X        }
X    } /* for */
X    closedir(dirp);
X    free(name);   /* done with this directory */
X
X    /* go do the rest */
X    return (ProcessDirs(nargc, nargv));
X}
X
X/*
X * Add a name to PruneList
X *
X */
XPRUNELIST *AddPrune(list, name)
XPRUNELIST *list;
Xchar *name;
X{
X    extern char *malloc();
X    register PRUNELIST *new;
X
X    if((new = (PRUNELIST *)malloc(sizeof(PRUNELIST))) == (PRUNELIST *)NULL)
X    {
X         /* outa mem */
X         perror(ProgName);
X         return(new);
X    }
X
X    new->name = name;
X    new->next = list;
X
X    return(new);
X}
X
```

```
X/*
X * Search for name on PruneList
X *
X */
Xint OnPruneList(list, name)
Xregister PRUNELIST *list;
Xregister char *name;
X{
X    for(; list != (PRUNELIST *)NULL; list = list->next)
X    {
X         if(strcmp(list->name, name) == 0)
X             return(~FALSE);
X    }
X
X    return(FALSE);
X}
X
X
X
X#ifdef TEST
X/* to test: cc -o name name.c -DTEST -DCASE; name ~ */
Xint MAIN(argc, argv)
Xint argc;
Xchar **argv;
X{
X    register int i;
X
X    for(i = 0; i < argc; i++)
X        printf("%d:\t%s\n", i, argv[i]);
X
X
X    return(0);
X}
X#endif /* TEST */
X
X#endif /* CASE */
SHAR_EOF
chmod 0600 expandar.c || echo "restore of expandar.c fails"
sed 's/^X//' << 'SHAR_EOF' > gz &&
Xsz -c "sz $*"
SHAR_EOF
chmod 0600 gz || echo "restore of gz fails"
sed 's/^X//' << 'SHAR_EOF' > minirb.1 &&
X'\" Revision Level
X'\" Last Delta     11-15-86
X.TH MINIRB 1 OMEN
X.SH NAME
Xminirb \- Bootstrap YMODEM Batch file receive
X.SH SYNOPSIS
X.B minirb
X.SH DESCRIPTION
X.I Minirb
Xreceives one or more files with YMODEM batch file transfer protocol.
```

```
X.I Minirb
Xwas developed for use as a bootstrap to simplify uploading of the longer
X.I rz
Xand
X.I sz
Xprograms contained in the
X.I rzsz1et2.sh
Xshell archive.
X.SH SEE ALSO
XYMODEM.DOC,
XProfessional-YAM manual,
Xsz(omen),
Xusq(omen),
Xundos(omen)
X.SH NOTES
XThe source file
X.I minirb.c
Xhas been ruthlessly pruned to simplify uploading without benefit of
Xany file transfer protocol.
XDo not reformat or add tabs to the source file
Xas this would complicate uploading.
X
XPlease read the uploading suggestions in the chapter
X.I "File Transfer Basics"
Xin the Professional-YAM or ZCOMM user's manual.
X
X.I Minirb
Xuses 8 bit checksum which may not be compatible with some programs
Xclaiming to support YMODEM batch transfers.
X
X.I Minirb
Xuses the
XYMODEM header
Xfile length information
Xto prevent extraneous characters.
X.I Minirb
Xdoes not delete carriage returns from uploaded files.
XIt does not delete ^Z or other padding added by some CP/M or PC-DOS
Xprograms, including the Xenix "doscp" command.
X
X.I Minirb
Xcalls stty(1) to set and reset raw tty mode.
SHAR_EOF
chmod 0600 minirb.1 || echo "restore of minirb.1 fails"
sed 's/^X//' << 'SHAR_EOF' > minirb.c &&
X/*
X * minirb.c By Chuck Forsberg Omen Technology INC
X *        "The High Reliability Communications Software"
X *
X * A bootstrap program for Unix to receive files from computers running
X *  YMODEM Batch (Professional-YAM, PowerCom, ZCOMM, etc.).
X *
X *  Minirb uses system(3) to call stty, avoiding system dependent code.
```

```
X *  Please refer to rz.c for comments, etc.
X */
Xchar * Version = "minirb 1.00 11-15-86";
X
X#include <stdio.h>
X#include <signal.h>
X#include <setjmp.h>
X
X#define OK 0
X#define FALSE 0
X#define TRUE 1
X#define ERROR (-1)
X#define CAN ('X'&037)
X#define SOH 1
X#define STX 2
X#define EOT 4
X#define ACK 6
X#define NAK 025
X#define TIMEOUT (-2)
X#define RETRYMAX 9
X#define WCEOT (-10)
X
XFILE *fout;
Xlong Bytesleft;
Xint Blklen;
Xchar secbuf[1024];
Xchar linbuf[1024];
Xint Lleft=0;
Xjmp_buf tohere;
X
Xalrm() { longjmp(tohere, -1); }
X
Xbibi(n) {
X canit(); mode(0);
X fprintf(stderr, "minirb: caught signal %d; exiting", n);
X exit(128+n);
X}
X
Xmode(n) {
X if (n) system("stty raw -echo");
X else system("stty echo -raw");
X}
X
Xmain() {
X mode(1);
X if (signal(SIGINT, bibi) == SIG_IGN) {
X  signal(SIGINT, SIG_IGN); signal(SIGKILL, SIG_IGN);
X } else {
X  signal(SIGINT, bibi); signal(SIGKILL, bibi);
X }
X#ifndef CASE
X printf("minirb: Now send file(s) with \042sb file ...\042 command\r\n");
X#endif
```

```
X
X if (wcreceive()==ERROR)
X  canit();
X mode(0); exit(0);
X}
X
Xwcreceive() {
X for (;;) {
X  if (wcrxpn(secbuf) == ERROR) break;
X  if (secbuf[0]==0) return OK;
X  if (procheader(secbuf)==ERROR || wcrx()==ERROR) break;
X }
X canit(); return ERROR;
X}
X
X
Xwcrxpn(rpn) char *rpn; {
X register c;
X
X purgeline();
Xet_tu:
X sendline(NAK); Lleft=0;
X while ((c = wcgetsec(rpn, 100)) != 0) {
X  if (c == WCEOT) { sendline(ACK); Lleft=0; readline(1); goto et_tu; }
X  return ERROR;
X }
X sendline(ACK); return OK;
X}
X
Xwcrx() {
X register int sectnum, sectcurr, sendchar, cblklen;
X
X sectnum=0; sendchar=NAK;
X for (;;) {
X  sendline(sendchar); Lleft=0;
X  sectcurr=wcgetsec(secbuf, (sectnum&0177)?50:130);
X  if (sectcurr==(sectnum+1)) {
X   sectnum++; cblklen = Bytesleft>Blklen ? Blklen:Bytesleft;
X   if (putsec(secbuf, cblklen)==ERROR) return ERROR;
X   if ((Bytesleft-=cblklen) < 0) Bytesleft = 0;
X   sendchar=ACK;
X  }
X  else if (sectcurr==sectnum) sendchar=ACK;
X  else if (sectcurr==WCEOT) {
X   if (fclose(fout)==ERROR) return ERROR;
X   sendline(ACK); Lleft=0; return OK;
X  }
X  else if (sectcurr==ERROR) return ERROR;
X  else return ERROR;
X }
X}
X
Xwcgetsec(rxbuf, maxtime) char *rxbuf; int maxtime; {
```

```
X register checksum, wcj, firstch; register char *p; int sectcurr, errors;
X for (errors=0; errors<RETRYMAX; errors++) {
X  if ((firstch=readline(maxtime))==STX) { Blklen=1024; goto get2; }
X  if (firstch==SOH) {
X   Blklen=128;
Xget2:
X   sectcurr=readline(1); checksum=0;
X   if ((sectcurr+(readline(1)))==0377) {
X    for (p=rxbuf,wcj=Blklen; --wcj>=0; ) {
X     if ((firstch=readline(1)) < 0) goto bilge;
X     checksum += (*p++ = firstch);
X    }
X    if ((firstch=readline(1)) < 0) goto bilge;
X    if (((checksum-firstch)&0377)==0) return sectcurr;
X   }
X  }
X  else if (firstch==EOT) return WCEOT;
X  else if (firstch==CAN) return ERROR;
Xbilge:
X  while(readline(1)!=TIMEOUT)
X   ;
X  maxtime=40; sendline(NAK); Lleft=0;
X }
X canit(); return ERROR;
X}
X
Xreadline(timeout) int timeout; {
X register n; static char *cdq;
X
X if (--Lleft >= 0) return (*cdq++ & 0377);
X n = timeout/10;
X if (n < 2) n = 3;
X if (setjmp(tohere)) { Lleft = 0; return TIMEOUT; }
X signal(SIGALRM, alrm); alarm(n);
X Lleft=read(0, cdq=linbuf, 1024); alarm(0);
X if (Lleft < 1) return TIMEOUT;
X --Lleft; return (*cdq++ & 0377);
X}
X
Xpurgeline() { Lleft = 0; lseek(0, 0L, 2); }
X
X
Xprocheader(name) char *name; {
X register char *p;
X
X Bytesleft = 2000000000L; p = name + 1 + strlen(name);
X if (*p) sscanf(p, "%ld", &Bytesleft);
X if ((fout=fopen(name, "w")) == NULL) return ERROR;
X return OK;
X}
X
Xputsec(buf, n) char *buf; register n;
X{
```

```
X register char *p;
X for (p=buf; --n>=0; ) putc( *p++, fout);
X return OK;
X}
X
Xsendline(c) { char d; d = c; write(1, &d, 1); }
X
Xchar canistr[] = { 24,24,24,24,24,24,24,24,0 };
X
Xcanit() { printf(canistr); Lleft=0; }
X
X/* END of minirb.c */
X
SHAR_EOF
chmod 0600 minirb.c || echo "restore of minirb.c fails"
sed 's/^X//' << 'SHAR_EOF' > rbsb.c &&
X/*% shar -f /tmp/rzsz README zmodem.h zm.c sz.c rz.c rbsb.c minirb.c *.1 gz
ptest.sh; shar -f /tmp/rzsz1et2 -m 1000000 zmodem.h zm.c sz.c rz.c rbsb.c minirb.c
*.1 gz ptest.sh
X *
X * -rev 03-10-87
X *  This file contains Unix specific stuff for setting terminal modes,
X *  very little is specific to ZMODEM or YMODEM per se (that stuff is in
X *  sz.c and rz.c).  The CRC-16 routines used by XMODEM, YMODEM, and ZMODEM
X *  are also in this file, a fast table driven macro version
X *
X *   This file is #included so the main file can set parameters such as HOWMANY.
X *   See the main files (rz.c/sz.c) for compile instructions.
X */
X
X#ifdef V7
X#include <sys/types.h>
X#include <sys/stat.h>
X#include <sgtty.h>
X#define OS "V7/BSD"
X#endif
X
X#ifndef OS
X#ifndef USG
X#define USG
X#endif
X#endif
X
X#ifdef USG
X#include <sys/types.h>
X#include <sys/stat.h>
X#include <termio.h>
X#include <sys/ioctl.h>
X#define OS "SYS III/V"
X#endif
X
X#if HOWMANY  > 255
XHowmany must be 255 or less
```

```
X#endif
X
Xstruct {
X    unsigned baudr;
X    int speedcode;
X} speeds[] = {
X    110, B110,
X    300, B300,
X    600, B600,
X    1200,     B1200,
X    2400,     B2400,
X    4800,     B4800,
X    9600,     B9600,
X    19200,    EXTA,
X    9600,     EXTB,
X    0,
X};
X
Xint Twostop;      /* Use two stop bits */
X
Xstatic unsigned
Xgetspeed(code)
X{
X    register n;
X
X    for (n=0; speeds[n].baudr; ++n)
X         if (speeds[n].speedcode == code)
X              return speeds[n].baudr;
X    return 0;
X}
X
X
X
X#ifdef ICANON
Xstruct termio oldtty, tty;
X#else
Xstruct sgttyb oldtty, tty;
Xstruct tchars oldtch, tch;
X#endif
X
Xint iofd = 0;          /* File descriptor for ioctls & reads */
X
X/*
X * mode(n)
X *  3: save old tty stat, set raw mode with flow control
X *  2: set a cbreak, XON/XOFF control mode if using Pro-YAM's -g option
X *  1: save old tty stat, set raw mode
X *  0: restore original tty mode
X */
Xmode(n)
X{
X    static did0 = FALSE;
X
```

```
X    vfile("mode:%d", n);
X    switch(n) {
X#ifdef USG
X    case 2:  /* Cbreak mode used by sb when -g detected */
X         if(!did0)
X              (void) ioctl(iofd, TCGETA, &oldtty);
X         tty = oldtty;
X
X         tty.c_iflag = BRKINT|IXON;
X
X         tty.c_oflag = 0;   /* Transparent output */
X
X         tty.c_cflag &= ~PARENB;/* Disable parity */
X         tty.c_cflag |= CS8;    /* Set character size = 8 */
X         if (Twostop)
X              tty.c_cflag |= CSTOPB;  /* Set two stop bits */
X
X         tty.c_lflag =
X#ifdef XCLUDE
X           XCLUDE |
X#endif
X           Zmodem ? 0 : ISIG;
X
X         tty.c_cc[VINTR] = 030;      /* Interrupt char */
X         tty.c_cc[VMIN] = 1;
X
X         (void) ioctl(iofd, TCSETAW, &tty);
X         did0 = TRUE;
X         return OK;
X    case 1:
X    case 3:
X         if(!did0)
X              (void) ioctl(iofd, TCGETA, &oldtty);
X         tty = oldtty;
X
X         tty.c_iflag = n==3 ? (IGNBRK|IXOFF) : IGNBRK;
X
X          /* No echo, crlf mapping, INTR, QUIT, delays, no erase/kill */
X         tty.c_lflag &= ~(ECHO | ICANON | ISIG);
X#ifdef XCLUDE
X         tty.c_lflag |= XCLUDE;
X#endif
X
X         tty.c_oflag = 0;   /* Transparent output */
X
X         tty.c_cflag &= ~PARENB;/* Same baud rate, disable parity */
X         tty.c_cflag |= CS8;    /* Set character size = 8 */
X         if (Twostop)
X              tty.c_cflag |= CSTOPB;  /* Set two stop bits */
X#ifdef NFGVMIN
X         tty.c_cc[VMIN] = 1; /* This many chars satisfies reads */
X#else
X         tty.c_cc[VMIN] = HOWMANY; /* This many chars satisfies reads */
```

```
X#endif
X        tty.c_cc[VTIME] = 1;    /* or in this many tenths of seconds */
X        (void) ioctl(iofd, TCSETAW, &tty);
X        did0 = TRUE;
X        Baudrate = getspeed(tty.c_cflag & CBAUD);
X        return OK;
X#endif
X#ifdef V7
X    case 2:  /*  This doesn't work ... */
X         printf("No mode(2) in V7/BSD!"); bibi(99);
X         if(!did0) {
X              ioctl(iofd, TIOCEXCL, 0);
X              ioctl(iofd, TIOCGETP, &oldtty);
X              ioctl(iofd, TIOCGETC, &oldtch);
X         }
X         tty = oldtty;
X         tch = oldtch;
X         tch.t_intrc = Zmodem ? 03:030;   /* Interrupt char */
X         tty.sg_flags |= (ODDP|EVENP|CBREAK);
X         tty.sg_flags &= ~(ALLDELAY|CRMOD|ECHO|LCASE);
X         ioctl(iofd, TIOCSETP, &tty);
X         ioctl(iofd, TIOCSETC, &tch);
X         did0 = TRUE;
X         return OK;
X    case 1:
X    case 3:
X         if(!did0) {
X              ioctl(iofd, TIOCEXCL, 0);
X              ioctl(iofd, TIOCGETP, &oldtty);
X              ioctl(iofd, TIOCGETC, &oldtch);
X         }
X         tty = oldtty;
X         tty.sg_flags |= RAW;
X         tty.sg_flags &= ~ECHO;
X         ioctl(iofd, TIOCSETP, &tty);
X         did0 = TRUE;
X         Baudrate = getspeed(tty.sg_ospeed);
X         return OK;
X#endif
X    case 0:
X         if(!did0)
X              return ERROR;
X#ifdef USG
X         (void) ioctl(iofd, TCSBRK, 1);   /* Wait for output to drain */
X         (void) ioctl(iofd, TCFLSH, 1);   /* Flush input queue */
X         (void) ioctl(iofd, TCSETAW, &oldtty);/* Restore original modes */
X         (void) ioctl(iofd, TCXONC,1);    /* Restart output */
X#endif
X#ifdef V7
X         ioctl(iofd, TIOCSETP, &oldtty);
X         ioctl(iofd, TIOCSETC, &oldtch);
X         ioctl(iofd, TIOCNXCL, 0);
X#endif
```

```
X         return OK;
X    default:
X         return ERROR;
X    }
X}
X
Xsendbrk()
X{
X#ifdef V7
X#ifdef TIOCSBRK
X#define CANBREAK
X    sleep(1);
X    ioctl(iofd, TIOCSBRK, 0);
X    sleep(1);
X    ioctl(iofd, TIOCCBRK, 0);
X#endif
X#endif
X#ifdef USG
X#define CANBREAK
X    ioctl(iofd, TCSBRK, 0);
X#endif
X}
X
X#ifdef FIONREAD
X#define READCHECK
X/*
X *  Return non 0 iff something to read from io descriptor f
X */
Xrdchk(f)
X{
X    static long lf;
X
X    ioctl(f, FIONREAD, &lf);
X    return ((int) lf);
X}
X#endif
X#ifdef SVR2
X#define READCHECK
X#include <fcntl.h>
X
Xchar checked = '\0' ;
X/*
X * Nonblocking I/O is a bit different in System V, Release 2
X */
Xrdchk(f)
X{
X    int lf, savestat = fcntl(f, F_GETFL) ;
X
X    fcntl(f, F_SETFL, savestat | O_NDELAY) ;
X    lf = read(f, &checked, 1) ;
X    fcntl(f, F_SETFL, savestat) ;
X    return(lf) ;
X}
```

```
X#endif
X
X
X/* crctab calculated by Mark G. Mendel, Network Systems Corporation */
Xstatic unsigned short crctab[256] = {
X    0x0000,  0x1021,  0x2042,  0x3063,  0x4084,  0x50a5,  0x60c6,  0x70e7,
X    0x8108,  0x9129,  0xa14a,  0xb16b,  0xc18c,  0xd1ad,  0xe1ce,  0xf1ef,
X    0x1231,  0x0210,  0x3273,  0x2252,  0x52b5,  0x4294,  0x72f7,  0x62d6,
X    0x9339,  0x8318,  0xb37b,  0xa35a,  0xd3bd,  0xc39c,  0xf3ff,  0xe3de,
X    0x2462,  0x3443,  0x0420,  0x1401,  0x64e6,  0x74c7,  0x44a4,  0x5485,
X    0xa56a,  0xb54b,  0x8528,  0x9509,  0xe5ee,  0xf5cf,  0xc5ac,  0xd58d,
X    0x3653,  0x2672,  0x1611,  0x0630,  0x76d7,  0x66f6,  0x5695,  0x46b4,
X    0xb75b,  0xa77a,  0x9719,  0x8738,  0xf7df,  0xe7fe,  0xd79d,  0xc7bc,
X    0x48c4,  0x58e5,  0x6886,  0x78a7,  0x0840,  0x1861,  0x2802,  0x3823,
X    0xc9cc,  0xd9ed,  0xe98e,  0xf9af,  0x8948,  0x9969,  0xa90a,  0xb92b,
X    0x5af5,  0x4ad4,  0x7ab7,  0x6a96,  0x1a71,  0x0a50,  0x3a33,  0x2a12,
X    0xdbfd,  0xcbdc,  0xfbbf,  0xeb9e,  0x9b79,  0x8b58,  0xbb3b,  0xab1a,
X    0x6ca6,  0x7c87,  0x4ce4,  0x5cc5,  0x2c22,  0x3c03,  0x0c60,  0x1c41,
X    0xedae,  0xfd8f,  0xcdec,  0xddcd,  0xad2a,  0xbd0b,  0x8d68,  0x9d49,
X    0x7e97,  0x6eb6,  0x5ed5,  0x4ef4,  0x3e13,  0x2e32,  0x1e51,  0x0e70,
X    0xff9f,  0xefbe,  0xdfdd,  0xcffc,  0xbf1b,  0xaf3a,  0x9f59,  0x8f78,
X    0x9188,  0x81a9,  0xb1ca,  0xa1eb,  0xd10c,  0xc12d,  0xf14e,  0xe16f,
X    0x1080,  0x00a1,  0x30c2,  0x20e3,  0x5004,  0x4025,  0x7046,  0x6067,
X    0x83b9,  0x9398,  0xa3fb,  0xb3da,  0xc33d,  0xd31c,  0xe37f,  0xf35e,
X    0x02b1,  0x1290,  0x22f3,  0x32d2,  0x4235,  0x5214,  0x6277,  0x7256,
X    0xb5ea,  0xa5cb,  0x95a8,  0x8589,  0xf56e,  0xe54f,  0xd52c,  0xc50d,
X    0x34e2,  0x24c3,  0x14a0,  0x0481,  0x7466,  0x6447,  0x5424,  0x4405,
X    0xa7db,  0xb7fa,  0x8799,  0x97b8,  0xe75f,  0xf77e,  0xc71d,  0xd73c,
X    0x26d3,  0x36f2,  0x0691,  0x16b0,  0x6657,  0x7676,  0x4615,  0x5634,
X    0xd94c,  0xc96d,  0xf90e,  0xe92f,  0x99c8,  0x89e9,  0xb98a,  0xa9ab,
X    0x5844,  0x4865,  0x7806,  0x6827,  0x18c0,  0x08e1,  0x3882,  0x28a3,
X    0xcb7d,  0xdb5c,  0xeb3f,  0xfb1e,  0x8bf9,  0x9bd8,  0xabbb,  0xbb9a,
X    0x4a75,  0x5a54,  0x6a37,  0x7a16,  0x0af1,  0x1ad0,  0x2ab3,  0x3a92,
X    0xfd2e,  0xed0f,  0xdd6c,  0xcd4d,  0xbdaa,  0xad8b,  0x9de8,  0x8dc9,
X    0x7c26,  0x6c07,  0x5c64,  0x4c45,  0x3ca2,  0x2c83,  0x1ce0,  0x0cc1,
X    0xef1f,  0xff3e,  0xcf5d,  0xdf7c,  0xaf9b,  0xbfba,  0x8fd9,  0x9ff8,
X    0x6e17,  0x7e36,  0x4e55,  0x5e74,  0x2e93,  0x3eb2,  0x0ed1,  0x1ef0
X};
X
X/*
X * updcrc macro derived from article Copyright (C) 1986 Stephen Satchell.
X *  NOTE: First srgument must be in range 0 to 255.
X *        Second argument is referenced twice.
X *
X * Programmers may incorporate any or all code into their programs,
X * giving proper credit within the source. Publication of the
X * source routines is permitted so long as proper credit is given
X * to Stephen Satchell, Satchell Evaluations and Chuck Forsberg,
SHAR_EOF
echo "End of part 1, continue with part 2"
echo "2" > s2_seq_.tmp
exit 0
```

From -2951658962344711151
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,226b7e5237351ed2,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-27 11:40:39 PST
Path: utzoo!attcan!uunet!seismo!sundc!pitstop!sun!decwrl!ucbvax!agate!saturn!ssyx!
koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i037: rzsz -- UNIX side for zmdm file transfer program part02/05
Keywords: UNIX
Message-ID: <3484@saturn.ucsc.edu>
Date: 27 May 88 18:40:39 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1052
Approved: koreth@ssyx.ucsc.edu

Submitted-by: bammi@mandrill.ces.cwru.edu (Jwahar R. Bammi)
Posting-number: Volume 1, Issue 37
Archive-name: rzsz/part02

```
#!/bin/sh
# this is part 2 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file rbsb.c continued
#
CurArch=2
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> rbsb.c
X * Omen Technology.
X */
X
X#define updcrc(cp, crc) ( crctab[((crc >> 8) & 255)] ^ (crc << 8) ^ cp)
X
X/*
X * Copyright (C) 1986 Gary S. Brown.  You may use this program, or
X * code or tables extracted from it, as desired without restriction.
X *
X *   Need an unsigned type capable of holding 32 bits;
X */
Xtypedef unsigned long int UNS_32_BITS;
X
X/* First, the polynomial itself and its table of feedback terms.  The  */
X/* polynomial is                                                       */
```

```
X/* X^32+X^26+X^23+X^22+X^16+X^12+X^11+X^10+X^8+X^7+X^5+X^4+X^2+X^1+X^0 */
X/* Note that we take it "backwards" and put the highest-order term in  */
X/* the lowest-order bit.  The X^32 term is "implied"; the LSB is the   */
X/* X^31 term, etc.  The X^0 term (usually shown as "+1") results in    */
X/* the MSB being 1.                                                    */
X
X/* Note that the usual hardware shift register implementation, which   */
X/* is what we're using (we're merely optimizing it by doing eight-bit  */
X/* chunks at a time) shifts bits into the lowest-order term.  In our   */
X/* implementation, that means shifting towards the right.  Why do we   */
X/* do it this way?  Because the calculated CRC must be transmitted in  */
X/* order from highest-order term to lowest-order term.  UARTs transmit */
X/* characters in order from LSB to MSB.  By storing the CRC this way,  */
X/* we hand it to the UART in the order low-byte to high-byte; the UART */
X/* sends each low-bit to hight-bit; and the result is transmission bit */
X/* by bit from highest- to lowest-order term without requiring any bit */
X/* shuffling on our part.  Reception works similarly.                  */
X
X/* The feedback terms table consists of 256, 32-bit entries.  Notes:   */
X/*                                                                     */
X/*     The table can be generated at runtime if desired; code to do so */
X/*     is shown later.  It might not be obvious, but the feedback      */
X/*     terms simply represent the results of eight shift/xor opera-    */
X/*     tions for all combinations of data and CRC register values.     */
X/*                                                                     */
X/*     The values must be right-shifted by eight bits by the "updcrc"  */
X/*     logic; the shift must be unsigned (bring in zeroes).  On some   */
X/*     hardware you could probably optimize the shift in assembler by  */
X/*     using byte-swap instructions.                                   */
X
Xstatic UNS_32_BITS crc_32_tab[] = { /* CRC polynomial 0xedb88320 */
X0x00000000, 0x77073096, 0xee0e612c, 0x990951ba, 0x076dc419, 0x706af48f,
0xe963a535, 0x9e6495a3,
X0x0edb8832, 0x79dcb8a4, 0xe0d5e91e, 0x97d2d988, 0x09b64c2b, 0x7eb17cbd,
0xe7b82d07, 0x90bf1d91,
X0x1db71064, 0x6ab020f2, 0xf3b97148, 0x84be41de, 0x1adad47d, 0x6ddde4eb,
0xf4d4b551, 0x83d385c7,
X0x136c9856, 0x646ba8c0, 0xfd62f97a, 0x8a65c9ec, 0x14015c4f, 0x63066cd9,
0xfa0f3d63, 0x8d080df5,
X0x3b6e20c8, 0x4c69105e, 0xd56041e4, 0xa2677172, 0x3c03e4d1, 0x4b04d447,
0xd20d85fd, 0xa50ab56b,
X0x35b5a8fa, 0x42b2986c, 0xdbbbc9d6, 0xacbcf940, 0x32d86ce3, 0x45df5c75,
0xdcd60dcf, 0xabd13d59,
X0x26d930ac, 0x51de003a, 0xc8d75180, 0xbfd06116, 0x21b4f4b5, 0x56b3c423,
0xcfba9599, 0xb8bda50f,
X0x2802b89e, 0x5f058808, 0xc60cd9b2, 0xb10be924, 0x2f6f7c87, 0x58684c11,
0xc1611dab, 0xb6662d3d,
X0x76dc4190, 0x01db7106, 0x98d220bc, 0xefd5102a, 0x71b18589, 0x06b6b51f,
0x9fbfe4a5, 0xe8b8d433,
X0x7807c9a2, 0x0f00f934, 0x9609a88e, 0xe10e9818, 0x7f6a0dbb, 0x086d3d2d,
0x91646c97, 0xe6635c01,
X0x6b6b51f4, 0x1c6c6162, 0x856530d8, 0xf262004e, 0x6c0695ed, 0x1b01a57b,
0x8208f4c1, 0xf50fc457,
```

```
X0x65b0d9c6, 0x12b7e950, 0x8bbeb8ea, 0xfcb9887c, 0x62dd1ddf, 0x15da2d49,
0x8cd37cf3, 0xfbd44c65,
X0x4db26158, 0x3ab551ce, 0xa3bc0074, 0xd4bb30e2, 0x4adfa541, 0x3dd895d7,
0xa4d1c46d, 0xd3d6f4fb,
X0x4369e96a, 0x346ed9fc, 0xad678846, 0xda60b8d0, 0x44042d73, 0x33031de5,
0xaa0a4c5f, 0xdd0d7cc9,
X0x5005713c, 0x270241aa, 0xbe0b1010, 0xc90c2086, 0x5768b525, 0x206f85b3,
0xb966d409, 0xce61e49f,
X0x5edef90e, 0x29d9c998, 0xb0d09822, 0xc7d7a8b4, 0x59b33d17, 0x2eb40d81,
0xb7bd5c3b, 0xc0ba6cad,
X0xedb88320, 0x9abfb3b6, 0x03b6e20c, 0x74b1d29a, 0xead54739, 0x9dd277af,
0x04db2615, 0x73dc1683,
X0xe3630b12, 0x94643b84, 0x0d6d6a3e, 0x7a6a5aa8, 0xe40ecf0b, 0x9309ff9d,
0x0a00ae27, 0x7d079eb1,
X0xf00f9344, 0x8708a3d2, 0x1e01f268, 0x6906c2fe, 0xf762575d, 0x806567cb,
0x196c3671, 0x6e6b06e7,
X0xfed41b76, 0x89d32be0, 0x10da7a5a, 0x67dd4acc, 0xf9b9df6f, 0x8ebeeff9,
0x17b7be43, 0x60b08ed5,
X0xd6d6a3e8, 0xa1d1937e, 0x38d8c2c4, 0x4fdff252, 0xd1bb67f1, 0xa6bc5767,
0x3fb506dd, 0x48b2364b,
X0xd80d2bda, 0xaf0a1b4c, 0x36034af6, 0x41047a60, 0xdf60efc3, 0xa867df55,
0x316e8eef, 0x4669be79,
X0xcb61b38c, 0xbc66831a, 0x256fd2a0, 0x5268e236, 0xcc0c7795, 0xbb0b4703,
0x220216b9, 0x5505262f,
X0xc5ba3bbe, 0xb2bd0b28, 0x2bb45a92, 0x5cb36a04, 0xc2d7ffa7, 0xb5d0cf31,
0x2cd99e8b, 0x5bdeae1d,
X0x9b64c2b0, 0xec63f226, 0x756aa39c, 0x026d930a, 0x9c0906a9, 0xeb0e363f,
0x72076785, 0x05005713,
X0x95bf4a82, 0xe2b87a14, 0x7bb12bae, 0x0cb61b38, 0x92d28e9b, 0xe5d5be0d,
0x7cdcefb7, 0x0bdbdf21,
X0x86d3d2d4, 0xf1d4e242, 0x68ddb3f8, 0x1fda836e, 0x81be16cd, 0xf6b9265b,
0x6fb077e1, 0x18b74777,
X0x88085ae6, 0xff0f6a70, 0x66063bca, 0x11010b5c, 0x8f659eff, 0xf862ae69,
0x616bffd3, 0x166ccf45,
X0xa00ae278, 0xd70dd2ee, 0x4e048354, 0x3903b3c2, 0xa7672661, 0xd06016f7,
0x4969474d, 0x3e6e77db,
X0xaed16a4a, 0xd9d65adc, 0x40df0b66, 0x37d83bf0, 0xa9bcae53, 0xdebb9ec5,
0x47b2cf7f, 0x30b5ffe9,
X0xbdbdf21c, 0xcabac28a, 0x53b39330, 0x24b4a3a6, 0xbad03605, 0xcdd70693,
0x54de5729, 0x23d967bf,
X0xb3667a2e, 0xc4614ab8, 0x5d681b02, 0x2a6f2b94, 0xb40bbe37, 0xc30c8ea1,
0x5a05df1b, 0x2d02ef8d
X};
X
X#define UPDC32(octet, crc) (crc_32_tab[((crc) ^ (octet)) & 0xff] ^ ((crc) >> 8))
X
X/* End of rbsb.c */
X
SHAR_EOF
chmod 0600 rbsb.c || echo "restore of rbsb.c fails"
sed 's/^X//' << 'SHAR_EOF' > rz.1 &&
X'\" Revision Level
X'\" Last Delta     02-10-87
```

```
X.TH RZ 1 OMEN
X.SH NAME
Xrb, rz \- XMODEM, YMODEM, ZMODEM (Batch) file receive
X.SH SYNOPSIS
X.B rz
X.RB [\- "\ +1abpqtuv" ]
X.br
X.B rb
X.RB [\- "\ +1abqtuv" ]
X.br
X.B rz
X.RB [\- "\ 1abcqtuv" ]
X.I file
X.br
X.B gz
X.I "file ..."
X.br
X.RB [ \- ][ v ] rzCOMMAND
X.SH DESCRIPTION
XThis program uses error correcting protocol to receive
Xfiles over a serial port from a variety of programs running under
XPC-DOS, CP/M,
X.SM Unix,
Xand other operating systems.
X
XThe first form of
X.I rz
X(Receive ZMODEM)
Xreceives files with the ZMODEM batch protocol.
XIf the sending program does not support ZMODEM,
X.I rz
Xsteps down to YMODEM protocol
Xafter 50 seconds.
XThis delay can be eliminated by calling the program as
X.I rb .
X
XWhen receiving with XMODEM or YMODEM,
X.I Rz
Xaccepts either standard 128 byte sectors or
X1024 byte sectors
X(YAM
X.B -k
Xoption).
XThe user should determine when
Xthe longer block length
Xactually improves throughput without causing problems.
X
XIf extended file information (file length, etc.)
Xis received,
Xthe file length controls the number of bytes written to
Xthe output dataset (YMODEM only),
Xand the modify time and file mode
X(iff non zero)
```

```
Xare set accordingly.
X
XIf no extended file information is received,
Xslashes in the pathname are changed to underscore,
Xand any trailing period in the pathname is eliminated.
XThis conversion is useful for files received from CP/M systems.
XWith YMODEM, each file name is converted to lower case
Xunless it contains one or more lower case letters.
X
X
XThe second form of
X.I rz
Xreceives a single
X.I file
Xwith XMODEM protocol.
XThe user must supply the file name to both sending and receiving programs.
X
X.I Gz
Xis a shell script which calls
X.I sz
Xto command a Pro-YAM or compatible program to transmit the specified files.
XPathnames used with
X.I gz
Xmust be escaped if they have special significance to the Unix shell.
X.br
XEXAMPLE:
Xgz "-1 C:*.c D:*.h"
X
X
XThe third form of
X.I rz
Xis invoked as
X.B rzCOMMAND
X(with an optional leading \- as generated by login(1)).
XFor each received file,
Xrz will pipe the file to ``COMMAND filename''
Xwhere filename is the name of the transmitted file
Xwith the file contents as standard input.
X
XEach file transfer is acknowledged when COMMAND exits with 0 status.
XA non zero exit status terminates transfers.
X
XA typical use for this form is
X.I rzrmail
Xwhich calls rmail(1)
Xto post mail to the user specified by the transmitted file name.
XFor example, sending the file "caf" from a PC-DOS system to
X.I rzrmail
Xon a
X.SM Unix
Xsystem
Xwould result in the contents of the DOS file "caf" being mailed to user "caf".
X
```

```
XOn some
X.SM Unix
Xsystems, the login directory must contain a link to
XCOMMAND as login sets SHELL=rsh which disallows absolute
Xpathnames.
XIf invoked with a leading ``v'',
X.I rz
Xwill report progress to /tmp/rzlog.
XThe following entry works for
X.SM Unix
X3.0:
X.ce
Xrzrmail::5:1::/bin:/usr/local/rzrmail
XIf the SHELL environment variable includes
X.I "rsh"
Xor
X.I "rksh"
X(restricted shell),
Xrz will not accept absolute pathnames
Xor references to a parent directory,
Xwill not modify an existing file, and
Xremoves any files received in error.
X
XIf
X.B rz
Xis invoked with stdout and stderr to different datasets,
XVerbose is set to 2, causing frame by frame progress reports
Xto stderr.
XThis may be disabled with the
X.B q
Xoption.
X
X.PP
XThe meanings of the available options are:
X.PP
X.PD 0
X.TP
X.B 1
XUse file descriptor 1 for ioctls and reads (Unix only).
XBy default, file descriptor 0 is used for ioctls and reads.
XThis option allows
X.B zz
Xto be used with the
X.I Professional-YAM
X.B $
Xcommand.
X.TP
X.B a
XConvert files to
X.SM Unix
Xconventions by stripping carriage returns and all characters
Xbeginning with the first Control Z (CP/M end of file).
X.TP
```

```
X.B b
XBinary
X(tell it like it is)
Xfile transfer override.
X.TP
X.B c
XRequest 16 bit CRC.
XXMODEM file transfers default to 8 bit checksum.
XYMODEM and ZMODEM normally use 16 bit CRC.
X.TP
X.B D
XOutput file data to /dev/null; for testing.
X.TP
X.B p
X(ZMODEM) Protect: skip file if destination file exists.
X.TP
X.B q
XQuiet suppresses verbosity.
X.TP
X.B "t tim"
XChange timeout to
X.I tim
Xtenths of seconds.
X.TP
X.B v
XVerbose
Xcauses a list of file
Xnames to be appended to
X/tmp/rzlog .
XMore v's generate more output.
X.PD
X.ne 6
X.SH EXAMPLES
X.RE
X(Pro-YAM command)
X.RS
X.I <ALT-2>
X.br
XPro-YAM Command:
X.I "sz *.h *.c"
X.br
X(This automatically invokes
X.I rz
Xon the connected system.)
X.RE
X.SH SEE ALSO
XZMODEM.DOC,
XYMODEM.DOC,
XIMP(CP/M),
XProfessional-YAM,
Xsz(omen),
Xusq(omen),
Xundos(omen)
```

```
X
XCompile time options required
Xfor various operating systems are described in the
Xsource file.
X.SH NOTES
XThe Unix "ulimit" parameter must be set high enough
Xto permit large file transfers.
X
XThe TTY input buffering on some systems may not allow long blocks
Xor streaming input, especially at high baud rates.
XThe Pro-YAM
X.B "zmodem l"
Xnumeric parameter may be set to a value between 64 and 1024 to limit the
Xburst length ("zmodem pl100").
X
X32 bit CRC code courtesy Gary S. Brown.
X.SH BUGS
XPathnames are restricted to 127 characters.
XIn XMODEM single file mode, the pathname given on the command line
Xis still processed as described above.
XThe ASCII option\'s CR/LF to NL translation merely deletes CR\'s;
Xundos(omen) performs a more intelligent translation.
X.SH "VMS VERSION"
XSome of the #includes with file names enclosed with angle brackets <>
Xmay need to have the angle brackets changed to "", or vice versa.
X
XThe VMS version does not set binary mode according to the incoming
Xfile type.
XNon binary file processing consists of stripping all characters beginning
Xwith CPMEOF (^Z).
X
XThe VMS version does not set the file time.
X
XAt high speeds,
XVMS sometimes loses incoming characters, resulting in retries
Xand degradation of throughput.
X
XThe mysterious
XVMS C Standard I/O Package and RMS may interact to modify
Xfile contents unexpectedly.
X
XThe VMS version does not support invocation as
X.B rzCOMMAND .
XZMODEM has not yet been implemented on the VMS version.
X.SH "ZMODEM CAPABILITIES"
X.I Rz
Xsupports incoming ZMODEM binary (-b), ASCII (-a),
Xprotect (-p),
Xand append (-+)
Xrequests, and ZMODEM command execution.
X.SH FILES
Xrz.c, rbsb.c, zm.c, zmodem.h source files.
X
```

```
X/tmp/rzlog stores debugging output generated with -vv option.
X.SH CWRU `ENHANCEMENTS'
XIf you compiled
X.I rz
Xwith the pre-processor symbol `CASE' defined, then when the
Xremote sends files with full path names (with
X.I sz -f
Xin Zmodem mode), all subdirectories that are needed to receive
Xthese files will be created, if they do not already exist. Note that
Xpathnames are always considered relative to the current directory,
Xso even if the remote sends a file with the pathname "/foo/bar", the
Xfile goes into "./foo/bar". If the directory "./foo" does not exist,
Xit will be created.
X.sp 1
XThe CWRU `enhancements' use Berkeley Unix specific code, and have been
Xtested only under BSD4.3. When making
X.I rz
Xplease make sure that you are using the appropriate flavor of Unix, before
Xdefining the pre processor symbol `CASE'.
SHAR_EOF
chmod 0600 rz.1 || echo "restore of rz.1 fails"
sed 's/^X//' << 'SHAR_EOF' > rz.c &&
X#define VERSION "1.18 02-18-87"
X#define PUBDIR "/usr/spool/uucppublic"
X
X/*% cc  -K -O -i % -o rz; size rz
X *  (above for Xenix SYS V 2.2 delta+)
X *
X * rz.c By Chuck Forsberg
X *
X *  cc -O rz.c -o rz       USG (3.0) Unix
X *  cc -O -DV7  rz.c -o rz      Unix V7, BSD 2.8 - 4.3
X *
X *  ln rz rb           For either system
X *
X *  ln rz /usr/bin/rzrmail      For remote mail.  Make this the
X *                     login shell. rzrmail then calls
X *                     rmail(1) to deliver mail.
X *
X *
X *  Unix is a trademark of Western Electric Company
X *
X * A program for Unix to receive files and commands from computers running
X *  Professional-YAM, PowerCom, YAM, IMP, or programs supporting XMODEM.
X *  rz uses Unix buffered input to reduce wasted CPU time.
X *
X * Iff the program is invoked by rzCOMMAND, output is piped to
X * "COMMAND filename"
X *
X *  Some systems (Venix, Coherent, Regulus) may not support tty raw mode
X *  read(2) the same way as Unix. ONEREAD must be defined to force one
X *  character reads for these systems. Added 7-01-84 CAF
X *
```

```
X *  Alarm signal handling changed to work with 4.2 BSD 7-15-84 CAF
X *
X *  NFGVMIN Updated 2-18-87 CAF for Xenix systems where c_cc[VMIN]
X *  doesn't work properly (even though it compiles without error!),
X *
X *  HOWMANY should be tuned for best performance
X *
X *  USG UNIX (3.0) ioctl conventions courtesy  Jeff Martin
X */
X#define LOGFILE "/tmp/rzlog"
X#define zperr vfile
X
X#include <stdio.h>
X#include <signal.h>
X#include <setjmp.h>
X#include <ctype.h>
XFILE *popen();
X
X#define OK 0
X#define FALSE 0
X#define TRUE 1
X#define ERROR (-1)
X
X/*
X * Max value for HOWMANY is 255.
X *   A larger value reduces system overhead but may evoke kernel bugs.
X *   133 corresponds to an XMODEM/CRC sector
X */
X#ifndef HOWMANY
X#define HOWMANY 133
X#endif
X
Xint Zmodem=0;          /* ZMODEM protocol requested */
Xint Nozmodem = 0; /* If invoked as "rb" */
Xunsigned Baudrate;
X#include "rbsb.c" /* most of the system dependent stuff here */
X
Xchar *substr();
XFILE *fout;
X#ifdef CASE
Xchar myfilebuf[16384];
X#endif
X
X
X/*
X * Routine to calculate the free bytes on the current file system
X *  ~0 means many free bytes (unknown)
X */
Xlong getfree()
X{
X    return(~0L);  /* many free bytes ... */
X}
X
```

```
X/* Ward Christensen / CP/M parameters - Don't change these! */
X#define ENQ 005
X#define CAN ('X'&037)
X#define XOFF ('s'&037)
X#define XON ('q'&037)
X#define SOH 1
X#define STX 2
X#define EOT 4
X#define ACK 6
X#define NAK 025
X#define CPMEOF 032
X#define WANTCRC 0103  /* send C not NAK to get crc not checksum */
X#define TIMEOUT (-2)
X#define RCDO (-3)
X#define ERRORMAX 5
X#define RETRYMAX 5
X#define WCEOT (-10)
X#define SECSIZ 128    /* cp/m's Magic Number record size */
X#define PATHLEN 257   /* ready for 4.2 bsd ? */
X#define KSIZE 1024    /* record size with k option */
X#define UNIXFILE 0x8000    /* happens to the the S_IFREG file mask bit for stat
*/
X
Xint Lastrx;
Xint Crcflg;
Xint Firstsec;
Xint Eofseen;      /* indicates cpm eof (^Z) has been received */
Xint errors;
Xint Restricted=0; /* restricted; no /.. or ../ in filenames */
X#ifdef ONEREAD
X/* Sorry, Regulus and some others don't work right in raw mode! */
Xint Readnum = 1;  /* Number of bytes to ask for in read() from modem */
X#else
Xint Readnum = HOWMANY; /* Number of bytes to ask for in read() from modem */
X#endif
X
X#define DEFBYTL 2000000000L     /* default rx file size */
Xlong Bytesleft;       /* number of bytes of incoming file left */
Xlong Modtime;         /* Unix style mod time for incoming file */
Xshort Filemode;       /* Unix style mode for incoming file */
Xchar Pathname[PATHLEN];
Xchar *Progname;       /* the name by which we were called */
X
Xint Batch=0;
Xint Wcsmask=0377;
Xint Topipe=0;
Xint MakeLCPathname=TRUE;   /* make received pathname lower case */
Xint Verbose=0;
Xint Quiet=0;      /* overrides logic that would otherwise set verbose */
Xint Nflag = 0;        /* Don't really transfer files */
Xint Rxbinary=FALSE;   /* receive all files in bin mode */
Xint Rxascii=FALSE;    /* receive files in ascii (translate) mode */
Xint Thisbinary;       /* current file is to be received in bin mode */
```

```
Xint Blklen;       /* record length of received packets */
Xchar secbuf[KSIZE];
Xchar linbuf[HOWMANY];
Xint Lleft=0;      /* number of characters in linbuf */
Xtime_t timep[2];
Xchar Lzmanag;          /* Local file management request */
Xchar zconv;       /* ZMODEM file conversion request */
Xchar zmanag;      /* ZMODEM file management request */
Xchar ztrans;      /* ZMODEM file transport request */
X
Xjmp_buf tohere;        /* For the interrupt on RX timeout */
X
X#include "zm.c"
X
Xint tryzhdrtype=ZRINIT;    /* Header type to send corresponding to Last rx close
*/
X
Xalrm()
X{
X    longjmp(tohere, -1);
X}
X
X/* called by signal interrupt or terminate to clean things up */
Xbibi(n)
X{
X    if (Zmodem)
X         zmputs(Attn);
X    canit(); mode(0);
X    fprintf(stderr, "rz: caught signal %d; exiting", n);
X    exit(128+n);
X}
X
Xmain(argc, argv)
Xint argc;
Xchar **argv;
X{
X    register char *cp;
X    register npats;
X    char *virgin, **patts;
X    char *getenv();
X    int exitcode;
X
X    Rxtimeout = 100;
X    setbuf(stderr, NULL);
X    if ((cp=getenv("SHELL")) && (substr(cp, "rsh") || substr(cp, "rksh")))
X         Restricted=TRUE;
X
X    chkinvok(virgin=argv[0]);   /* if called as [-]rzCOMMAND set flag */
X    npats = 0;
X    while ((--argc) > 0) {
X         cp = *++argv;
X         if (*cp == '-') {
X              while( *++cp) {
```

```
X                    switch(*cp) {
X                    case '+':
X                         Lzmanag = ZMAPND; break;
X                    case '1':
X                         iofd = 1; break;
X                    case '7':
X                         Wcsmask = 0177;
X                    case 'a':
X                         Rxascii=TRUE;  break;
X                    case 'b':
X                         Rxbinary=TRUE; break;
X                    case 'c':
X                         Crcflg=TRUE; break;
X                    case 'D':
X                         Nflag = TRUE; break;
X                    case 'p':
X                         Lzmanag = ZMPROT;  break;
X                    case 'q':
X                         Quiet=TRUE; Verbose=0; break;
X                    case 't':
X                         if (--argc < 1) {
X                              usage();
X                         }
X                         Rxtimeout = atoi(*++argv);
X                         if (Rxtimeout<10 || Rxtimeout>1000)
X                              usage();
X                         break;
X                    case 'u':
X                         MakeLCPathname=FALSE; break;
X                    case 'v':
X                         ++Verbose; break;
X                    default:
X                         usage();
X                    }
X               }
X          }
X          else if ( !npats && argc>0) {
X               if (argv[0][0]) {
X                    npats=argc;
X                    patts=argv;
X               }
X          }
X     }
X     if (npats > 1)
X          usage();
X     if (Verbose) {
X          if (freopen(LOGFILE, "a", stderr)==NULL) {
X               printf("Can't open log file %s\n",LOGFILE);
X               exit(0200);
X          }
X          setbuf(stderr, NULL);
X          fprintf(stderr, "argv[0]=%s Progname=%s\n", virgin, Progname);
X     }
```

```
X    if (fromcu() && !Quiet) {
X         if (Verbose == 0)
X              Verbose = 2;
X    }
X    mode(1);
X    if (signal(SIGINT, bibi) == SIG_IGN) {
X         signal(SIGINT, SIG_IGN); signal(SIGKILL, SIG_IGN);
X    }
X    else {
X         signal(SIGINT, bibi); signal(SIGKILL, bibi);
X    }
X    if (wcreceive(npats, patts)==ERROR) {
X         exitcode=0200;
X         canit();
X    }
X    mode(0);
X    if (exitcode && !Zmodem)    /* bellow again with all thy might. */
X         canit();
X    exit(exitcode);
X}
X
X
Xusage()
X{
X    fprintf(stderr,"%s %s for %s by Chuck Forsberg\n",
X      Progname, VERSION, OS);
X    fprintf(stderr,"Usage:  rz [-1abuv]        (ZMODEM Batch)\n");
X    fprintf(stderr,"or rb [-1abuv]        (YMODEM Batch)\n");
X    fprintf(stderr,"or rz [-1abcv] file   (XMODEM or XMODEM-1k)\n");
X    fprintf(stderr,"     -1 For cu(1): Use fd 1 for input\n");
X    fprintf(stderr,"     -a ASCII transfer (strip CR)\n");
X    fprintf(stderr,"     -b Binary transfer for all files\n");
X    fprintf(stderr,"     -v Verbose more v's give more info\n");
X    fprintf(stderr,"     -c Use 16 bit CRC     (XMODEM)\n");
X    exit(1);
X}
X/*
X *  Debugging information output interface routine
X */
X/* VARARGS1 */
Xvfile(f, a, b, c)
Xregister char *f;
X{
X    if (Verbose > 1) {
X         fprintf(stderr, f, a, b, c);
X         fprintf(stderr, "\n");
X    }
X}
X
X/*
X * Let's receive something already.
X */
X
```

```
Xchar *rbmsg =
X"%s ready. To begin transfer, type \"%s file ...\" to your modem program\r\n";
X
Xwcreceive(argc, argp)
Xchar **argp;
X{
X    register c;
X
X    if (Batch || argc==0) {
X         Crcflg=(Wcsmask==0377);
X         if ( !Quiet)
X              fprintf(stderr, rbmsg, Progname, Nozmodem?"sb":"sz");
X         if (c=tryz()) {
X              if (c == ZCOMPL)
X                   return OK;
X              if (c == ERROR)
X                   goto fubar;
X              c = rzfiles();
X              if (c)
X                   goto fubar;
X         } else {
X              for (;;) {
X                   if (wcrxpn(secbuf)== ERROR)
X                        goto fubar;
X                   if (secbuf[0]==0)
X                        return OK;
X                   if (procheader(secbuf) == ERROR)
X                        goto fubar;
X                   if (wcrx()==ERROR)
X                        goto fubar;
X              }
X         }
X    } else {
X         Bytesleft = DEFBYTL; Filemode = 0; Modtime = 0L;
X         Thisbinary = Rxbinary; /* ++Jrb */
X
X         /* procheader(""); */
X         strcpy(Pathname, *argp); checkpath(Pathname);
X#ifdef CASE
X         if(Verbose)
X             fprintf(stderr, "\nrz: ready to receive %s [%s mode]\r\n",
X              Pathname, Thisbinary?"BIN":"ASCII");
X#endif
X
X         if ((fout=fopen(Pathname, "w")) == NULL)
X              return ERROR;
X#ifdef CASE
X         setbuffer(fout, myfilebuf, sizeof(myfilebuf));
X#endif
X         if (wcrx()==ERROR)
X              goto fubar;
X    }
X    return OK;
```

```
Xfubar:
X    canit();
X    if (Topipe && fout) {
X         pclose(fout);  return ERROR;
X    }
X    if (fout)
X         fclose(fout);
X    if (Restricted) {
X         unlink(Pathname);
X         fprintf(stderr, "\r\nrz: %s removed.\r\n", Pathname);
X    }
X    return ERROR;
X}
X
X
X/*
X * Fetch a pathname from the other end as a C ctyle ASCIZ string.
X * Length is indeterminate as long as less than Blklen
X * A null string represents no more files (YMODEM)
X */
Xwcrxpn(rpn)
Xchar *rpn;   /* receive a pathname */
X{
X    register c;
X
X#ifdef NFGVMIN
X    readline(1);
X#else
X    purgeline();
X#endif
X
Xet_tu:
X    Firstsec=TRUE;  Eofseen=FALSE;
X    sendline(Crcflg?WANTCRC:NAK);
X    Lleft=0;  /* Do read next time ... */
X    while ((c = wcgetsec(rpn, 100)) != 0) {
X         log( "Pathname fetch returned %d\n", c);
X         if (c == WCEOT) {
X              sendline(ACK);
X              Lleft=0;  /* Do read next time ... */
X              readline(1);
X              goto et_tu;
X         }
X         return ERROR;
X    }
X    sendline(ACK);
X    return OK;
X}
X
X/*
X * Adapted from CMODEM13.C, written by
X * Jack M. Wierda and Roderick W. Hart
X */
```

```
X
Xwcrx()
X{
X    register int sectnum, sectcurr;
X    register char sendchar;
X    register char *p;
X    int cblklen;           /* bytes to dump this block */
X
X    Firstsec=TRUE;sectnum=0; Eofseen=FALSE;
X    sendchar=Crcflg?WANTCRC:NAK;
X
X    for (;;) {
X         sendline(sendchar);    /* send it now, we're ready! */
X         Lleft=0; /* Do read next time ... */
X         sectcurr=wcgetsec(secbuf, (sectnum&0177)?50:130);
X         report(sectcurr);
X         if (sectcurr==(sectnum+1 &Wcsmask)) {
X              sectnum++;
X              cblklen = Bytesleft>Blklen ? Blklen:Bytesleft;
X              if (putsec(secbuf, cblklen)==ERROR)
X                   return ERROR;
X              if ((Bytesleft-=cblklen) < 0)
X                   Bytesleft = 0;
X              sendchar=ACK;
X         }
X         else if (sectcurr==(sectnum&Wcsmask)) {
X              log( "Received dup Sector\n");
X              sendchar=ACK;
X         }
X         else if (sectcurr==WCEOT) {
X              if (closeit())
X                   return ERROR;
X              sendline(ACK);
X              Lleft=0; /* Do read next time ... */
X              return OK;
X         }
X         else if (sectcurr==ERROR)
X              return ERROR;
X         else {
X              log( "Sync Error\n");
X              return ERROR;
X         }
X    }
X}
X
X/*
X * Wcgetsec fetches a Ward Christensen type sector.
X * Returns sector number encountered or ERROR if valid sector not received,
X * or CAN CAN received
X * or WCEOT if eot sector
X * time is timeout for first char, set to 4 seconds thereafter
X ***************** NO ACK IS SENT IF SECTOR IS RECEIVED OK **************
X *    (Caller must do that when he is good and ready to get next sector)
```

```
X */
X
Xwcgetsec(rxbuf, maxtime)
Xchar *rxbuf;
Xint maxtime;
X{
X    register checksum, wcj, firstch;
X    register unsigned short oldcrc;
X    register char *p;
X    int sectcurr;
X
X    for (Lastrx=errors=0; errors<RETRYMAX; errors++) {
X
X         if ((firstch=readline(maxtime))==STX) {
X              Blklen=KSIZE; goto get2;
X         }
X         if (firstch==SOH) {
X              Blklen=SECSIZ;
Xget2:
X              sectcurr=readline(1);
X              if ((sectcurr+(oldcrc=readline(1)))==Wcsmask) {
X                   oldcrc=checksum=0;
X                   for (p=rxbuf,wcj=Blklen; --wcj>=0; ) {
X                        if ((firstch=readline(1)) < 0)
X                             goto bilge;
X                        oldcrc=updcrc(firstch, oldcrc);
X                        checksum += (*p++ = firstch);
X                   }
X                   if ((firstch=readline(1)) < 0)
X                        goto bilge;
X                   if (Crcflg) {
X                        oldcrc=updcrc(firstch, oldcrc);
X                        if ((firstch=readline(1)) < 0)
X                             goto bilge;
X                        oldcrc=updcrc(firstch, oldcrc);
X                        if (oldcrc & 0xFFFF)
X                             log("CRC=0%o\n", oldcrc);
X                        else {
X                             Firstsec=FALSE;
X                             return sectcurr;
X                        }
X                   }
X                   else if (((checksum-firstch)&Wcsmask)==0) {
X                        Firstsec=FALSE;
X                        return sectcurr;
X                   }
X                   else
X                        log( "Checksum Error\n");
X              }
X              else
X                   log("Sector number garbled 0%o 0%o\n",
X                    sectcurr, oldcrc);
X         }
```

```
X         /* make sure eot really is eot and not just mixmash */
X#ifdef NFGVMIN
X         else if (firstch==EOT && readline(1)==TIMEOUT)
X              return WCEOT;
X#else
X         else if (firstch==EOT && Lleft==0)
X              return WCEOT;
X#endif
X         else if (firstch==CAN) {
X              if (Lastrx==CAN) {
X                   log( "Sender CANcelled\n");
X                   return ERROR;
X              } else {
X                   Lastrx=CAN;
X                   continue;
X              }
X         }
X         else if (firstch==TIMEOUT) {
X              if (Firstsec)
X                   goto humbug;
Xbilge:
X              log( "Timeout\n");
X         }
X         else
X              log( "Got 0%o sector header\n", firstch);
X
Xhumbug:
X         Lastrx=0;
X         while(readline(1)!=TIMEOUT)
X              ;
X         if (Firstsec) {
X              sendline(Crcflg?WANTCRC:NAK);
X              Lleft=0;  /* Do read next time ... */
X         } else {
X              maxtime=40; sendline(NAK);
X              Lleft=0;  /* Do read next time ... */
X         }
X    }
X    /* try to stop the bubble machine. */
X    canit();
X    return ERROR;
X}
X
X/*
X * This version of readline is reasoably well suited for
X * reading many characters.
X *  (except, currently, for the Regulus version!)
X *
X * timeout is in tenths of seconds
X */
Xreadline(timeout)
Xint timeout;
X{
```

```
X    register n;
X    static char *cdq;  /* pointer for removing chars from linbuf */
X
X    if (--Lleft >= 0) {
X#ifdef RDEBUG
X        if (Verbose > 8) {
X            fprintf(stderr, "%02x ", *cdq&0377);
X        }
X#endif
X        return (*cdq++ & Wcsmask);
X    }
X    n = timeout/10;
X    if (n < 2)
X        n = 3;
X#ifdef RDEBUG
X    if (Verbose > 3)
X        fprintf(stderr, "Calling read: alarm=%d  Readnum=%d ",
X          n, Readnum);
X#endif
X    if (setjmp(tohere)) {
X#ifdef TIOCFLUSH
X/*      ioctl(iofd, TIOCFLUSH, 0); */
X#endif
X        Lleft = 0;
X        if (Verbose>1)
X            fprintf(stderr, "Readline:TIMEOUT\n");
X        return TIMEOUT;
X    }
X    signal(SIGALRM, alrm); alarm(n);
X    Lleft=read(iofd, cdq=linbuf, Readnum);
X    alarm(0);
X#ifdef RDEBUG
X    if (Verbose > 3) {
X        fprintf(stderr, "Read returned %d bytes\n", Lleft);
X    }
X#endif
X    if (Lleft < 1)
X        return TIMEOUT;
X    --Lleft;
X    if (Verbose > 8) {
X        fprintf(stderr, "%02x ", *cdq&0377);
X    }
X    return (*cdq++ & Wcsmask);
X}
X
X
X
X/*
X * Purge the modem input queue of all characters
X */
Xpurgeline()
X{
X    Lleft = 0;
```

```
X#ifdef USG
X    ioctl(iofd, TCFLSH, 0);
X#else
X    lseek(iofd, 0L, 2);
X#endif
X}
X
X
X/*
X * Process incoming file information header
X */
Xprocheader(name)
Xchar *name;
X{
X    register char *openmode, *p, **pp;
X
X    /* set default parameters and overrides */
X    openmode = "w";
X    Thisbinary = (!Rxascii) || Rxbinary;
X    if (Lzmanag)
X         zmanag = Lzmanag;
X
X    /*
X     *  Process ZMODEM remote file management requests
X     */
X    if (!Rxbinary && zconv == ZCNL)  /* Remote ASCII override */
X         Thisbinary = 0;
X    if (zconv == ZCBIN)    /* Remote Binary override */
X         Thisbinary = TRUE;
X    else if (zmanag == ZMAPND)
X         openmode = "a";
X    /* ZMPROT check for existing file */
X    if (zmanag == ZMPROT && (fout=fopen(name, "r"))) {
X         fclose(fout);  return ERROR;
X    }
X
X#ifdef CASE
X/* CWRU NOTE:
X *  We will not accept rooted paths ie. paths that begin in '/' or './'
X *  If the incoming filename is rooted, we skip the beginning
SHAR_EOF
echo "End of part 2, continue with part 3"
echo "3" > s2_seq_.tmp
exit 0


From 6215800908680359602
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,577d7faea88c712a,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-27 11:41:55 PST
Path: utzoo!attcan!uunet!seismo!sundc!pitstop!sun!decwrl!ucbvax!agate!saturn!ssyx!
koreth
```

From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i038: rzsz -- UNIX side for zmdm file transfer program part03/05
Keywords: UNIX
Message-ID: <3485@saturn.ucsc.edu>
Date: 27 May 88 18:41:55 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1305
Approved: koreth@ssyx.ucsc.edu

Submitted-by: bammi@mandrill.ces.cwru.edu (Jwahar R. Bammi)
Posting-number: Volume 1, Issue 38
Archive-name: rzsz/part03

```
#!/bin/sh
# this is part 3 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file rz.c continued
#
CurArch=3
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> rz.c
X *  '/'   './'  or  '../'. If you want to allow rooted paths just
X *  remove the next block. NOTE that the code does not handle
X *  the '../../foo' case, which is unlikely to occur as normally
X *  you would be sending (with -f) from using something like
X *  sz -f foo/* or similar reg. expressions.
X */
X
X    if( (name[0] == '/') || (name[0] == '.')  )
X    {
X         /* skip over the leading stuff */
X         if(name[0] == '/')
X              name = &name[1];
X         else
X         {
X              if((name[1] == '.') && (name[2] == '/'))
X                   name = &name[3];  /* Skip the "../" */
X              else
X                  if(name[1] == '/')
X                   name = &name[2];  /* Skip the "./"  */
X         }
X    }
X
X     /* CWRU addition, create any dierctories in the path that don't exist */
```

```
X    if( pathensure(name) == ERROR)
X         return ERROR;
X#endif
X    Bytesleft = DEFBYTL; Filemode = 0; Modtime = 0L;
X
X    p = name + 1 + strlen(name);
X    if (*p) {/* file coming from Unix or DOS system */
X         sscanf(p, "%ld%lo%o", &Bytesleft, &Modtime, &Filemode);
X         if (Filemode & UNIXFILE)
X              ++Thisbinary;
X         if (Verbose) {
X              fprintf(stderr,  "Incoming: %s %ld %lo %o\n",
X                name, Bytesleft, Modtime, Filemode);
X         }
X    }
X    else {        /* File coming from CP/M system */
X         for (p=name; *p; ++p)       /* change / to _ */
X              if ( *p == '/')
X                   *p = '_';
X
X         if ( *--p == '.')      /* zap trailing period */
X              *p = 0;
X    }
X
X    if (!Zmodem && MakeLCPathname && !IsAnyLower(name))
X         uncaps(name);
X    if (Topipe) {
X         sprintf(Pathname, "%s %s", Progname+2, name);
X         if (Verbose)
X              fprintf(stderr,  "Topipe: %s %s\n",
X                Pathname, Thisbinary?"BIN":"ASCII");
X         if ((fout=popen(Pathname, "w")) == NULL)
X              return ERROR;
X#ifdef CASE
X         setbuffer(fout, myfilebuf, sizeof(myfilebuf));
X#endif
X    } else {
X         strcpy(Pathname, name);
X         if (Verbose) {
X              fprintf(stderr,  "Receiving %s %s %s\n",
X                name, Thisbinary?"BIN":"ASCII", openmode);
X         }
X         checkpath(name);
X         if (Nflag)
X              name = "/dev/null";
X         if ((fout=fopen(name, openmode)) == NULL)
X              return ERROR;
X    }
X    return OK;
X}
X
X/*
X * Putsec writes the n characters of buf to receive file fout.
```

```
X *  If not in binary mode, carriage returns, and all characters
X *  starting with CPMEOF are discarded.
X */
Xputsec(buf, n)
Xchar *buf;
Xregister n;
X{
X    register char *p;
X
X    if (Thisbinary) {
X         for (p=buf; --n>=0; )
X              putc( *p++, fout);
X    }
X    else {
X         if (Eofseen)
X              return OK;
X         for (p=buf; --n>=0; ++p ) {
X              if ( *p == '\r')
X                   continue;
X              if (*p == CPMEOF) {
X                   Eofseen=TRUE; return OK;
X              }
X              putc(*p ,fout);
X         }
X    }
X    return OK;
X}
X
X/*
X *  Send a character to modem.  Small is beautiful.
X */
Xsendline(c)
X{
X    char d;
X
X    d = c;
X    if (Verbose>4)
X         fprintf(stderr, "Sendline: %x\n", c);
X    write(1, &d, 1);
X}
X
Xxsendline(c)
X{
X    sendline(c);
X}
X
Xflushmo() {}
X
X
X
X
X/* make string s lower case */
Xuncaps(s)
```

```
Xregister char *s;
X{
X    for ( ; *s; ++s)
X         if (isupper(*s))
X              *s = tolower(*s);
X}
X/*
X * IsAnyLower returns TRUE if string s has lower case letters.
X */
XIsAnyLower(s)
Xregister char *s;
X{
X    for ( ; *s; ++s)
X         if (islower(*s))
X              return TRUE;
X    return FALSE;
X}
X
X/*
X * substr(string, token) searches for token in string s
X * returns pointer to token within string if found, NULL otherwise
X */
Xchar *
Xsubstr(s, t)
Xregister char *s,*t;
X{
X    register char *ss,*tt;
X    /* search for first char of token */
X    for (ss=s; *s; s++)
X         if (*s == *t)
X              /* compare token with substring */
X              for (ss=s,tt=t; ;) {
X                   if (*tt == 0)
X                        return s;
X                   if (*ss++ != *tt++)
X                        break;
X              }
X    return NULL;
X}
X
X/*
X * Log an error
X */
X/*VARARGS1*/
Xlog(s,p,u)
Xchar *s, *p, *u;
X{
X    if (!Verbose)
X         return;
X    fprintf(stderr, "error %d: ", errors);
X    fprintf(stderr, s, p, u);
X}
X
```

```
X/* send cancel string to get the other end to shut up */
Xcanit()
X{
X    static char canistr[] = {
X     24,24,24,24,24,24,24,24,24,8,8,8,8,8,8,8,8,8,0
X    };
X
X    printf(canistr);
X    Lleft=0;  /* Do read next time ... */
X    fflush(stdout);
X}
X
X
X/*
X * Return 1 iff stdout and stderr are different devices
X *  indicating this program operating with a modem on a
X *  different line
X */
Xfromcu()
X{
X    struct stat a, b;
X    fstat(1, &a); fstat(2, &b);
X    return (a.st_rdev != b.st_rdev);
X}
X
Xreport(sct)
Xint sct;
X{
X    if (Verbose>1)
X         fprintf(stderr,"%03d%c",sct,sct%10? ' ' : '\r');
X}
X
X/*
X * If called as [-][dir/../]vrzCOMMAND set Verbose to 1
X * If called as [-][dir/../]rzCOMMAND set the pipe flag
X * If called as rb use YMODEM protocol
X */
Xchkinvok(s)
Xchar *s;
X{
X    register char *p;
X
X    p = s;
X    while (*p == '-')
X         s = ++p;
X    while (*p)
X         if (*p++ == '/')
X              s = p;
X    if (*s == 'v') {
X         Verbose=1; ++s;
X    }
X    Progname = s;
X    if (s[0]=='r' && s[1]=='b')
```

```
X         Nozmodem = TRUE;
X    if (s[2] && s[0]=='r' && s[1]=='b')
X         Topipe=TRUE;
X    if (s[2] && s[0]=='r' && s[1]=='z')
X         Topipe=TRUE;
X}
X
X/*
X * Totalitarian Communist pathname processing
X */
Xcheckpath(name)
Xchar *name;
X{
X    if (Restricted) {
X         if (fopen(name, "r") != NULL) {
X              canit();
X              fprintf(stderr, "\r\nrz: %s exists\n", name);
X              bibi();
X         }
X         /* restrict pathnames to current tree or uucppublic */
X         if ( substr(name, "../")
X          || (name[0]== '/' && strncmp(name, PUBDIR, strlen(PUBDIR))) ) {
X              canit();
X              fprintf(stderr,"\r\nrz:\tSecurity Violation\r\n");
X              bibi();
X         }
X    }
X}
X
X/*
X * Initialize for Zmodem receive attempt, try to activate Zmodem sender
X *  Handles ZSINIT frame
X *  Return ZFILE if Zmodem filename received, -1 on error,
X *   ZCOMPL if transaction finished,  else 0
X */
Xtryz()
X{
X    register c, n;
X    register cmdzack1flg;
X
X    if (Nozmodem)      /* Check for "rb" program name */
X         return 0;
X
X
X    for (n=Zmodem?10:5; --n>=0; ) {
X         /* Set buffer length (0) and capability flags */
X         stohdr(0L);
X#ifdef CANBREAK
X         Txhdr[ZF0] = CANFC32|CANFDX|CANOVIO|CANBRK;
X#else
X         Txhdr[ZF0] = CANFC32|CANFDX|CANOVIO;
X#endif
X         zshhdr(tryzhdrtype, Txhdr);
```

```
Xagain:
X	switch (zgethdr(Rxhdr, 0)) {
X	case ZRQINIT:
X		continue;
X	case ZEOF:
X		continue;
X	case TIMEOUT:
X		continue;
X	case ZFILE:
X		zconv = Rxhdr[ZF0];
X		zmanag = Rxhdr[ZF1];
X		ztrans = Rxhdr[ZF2];
X		tryzhdrtype = ZRINIT;
X		c = zrdata(secbuf, KSIZE);
X		mode(3);
X		if (c == GOTCRCW)
X			return ZFILE;
X		zshhdr(ZNAK, Txhdr);
X		goto again;
X	case ZSINIT:
X		if (zrdata(Attn, ZATTNLEN) == GOTCRCW) {
X			zshhdr(ZACK, Txhdr);
X			goto again;
X		}
X		zshhdr(ZNAK, Txhdr);
X		goto again;
X	case ZFREECNT:
X		stohdr(getfree());
X		zshhdr(ZACK, Txhdr);
X		goto again;
X	case ZCOMMAND:
X		cmdzack1flg = Rxhdr[ZF0];
X		if (zrdata(secbuf, KSIZE) == GOTCRCW) {
X			if (cmdzack1flg & ZCACK1)
X				stohdr(0L);
X			else
X				stohdr((long)sys2(secbuf));
X			purgeline();  /* dump impatient questions */
X			do {
X				zshhdr(ZCOMPL, Txhdr);
X			}
X			while (++errors<10 && zgethdr(Rxhdr,1) != ZFIN);
X			ackbibi();
X			if (cmdzack1flg & ZCACK1)
X				exec2(secbuf);
X			return ZCOMPL;
X		}
X		zshhdr(ZNAK, Txhdr); goto again;
X	case ZCOMPL:
X		goto again;
X	default:
X		continue;
X	case ZFIN:
```

```
X             ackbibi(); return ZCOMPL;
X        case ZCAN:
X             return ERROR;
X        }
X    }
X    return 0;
X}
X
X/*
X * Receive 1 or more files with ZMODEM protocol
X */
Xrzfiles()
X{
X    register c;
X
X    for (;;) {
X         switch (c = rzfile()) {
X         case ZEOF:
X         case ZSKIP:
X              switch (tryz()) {
X              case ZCOMPL:
X                   return OK;
X              default:
X                   return ERROR;
X              case ZFILE:
X                   break;
X              }
X              continue;
X         default:
X              return c;
X         case ERROR:
X              return ERROR;
X         }
X    }
X}
X
X/*
X * Receive a file with ZMODEM protocol
X *  Assumes file name frame is in secbuf
X */
Xrzfile()
X{
X    register c, n;
X    long rxbytes;
X
X    Eofseen=FALSE;
X    if (procheader(secbuf) == ERROR) {
X         return (tryzhdrtype = ZSKIP);
X    }
X
X    n = 10; rxbytes = 0l;
X
X    for (;;) {
```

```
X         stohdr(rxbytes);
X         zshhdr(ZRPOS, Txhdr);
Xnxthdr:
X         switch (c = zgethdr(Rxhdr, 0)) {
X         default:
X              vfile("rzfile: zgethdr returned %d", c);
X              return ERROR;
X         case ZNAK:
X         case TIMEOUT:
X              if ( --n < 0) {
X                   vfile("rzfile: zgethdr returned %d", c);
X                   return ERROR;
X              }
X         case ZFILE:
X              zrdata(secbuf, KSIZE);
X              continue;
X         case ZEOF:
X              if (rclhdr(Rxhdr) != rxbytes) {
X                   continue;
X              }
X              if (closeit()) {
X                   tryzhdrtype = ZFERR;
X                   vfile("rzfile: closeit returned <> 0");
X                   return ERROR;
X              }
X              vfile("rzfile: normal EOF");
X              return c;
X         case ERROR:   /* Too much garbage in header search error */
X              if ( --n < 0) {
X                   vfile("rzfile: zgethdr returned %d", c);
X                   return ERROR;
X              }
X              zmputs(Attn);
X              continue;
X         case ZDATA:
X              if (rclhdr(Rxhdr) != rxbytes) {
X                   if ( --n < 0) {
X                        return ERROR;
X                   }
X                   zmputs(Attn);  continue;
X              }
Xmoredata:
X              switch (c = zrdata(secbuf, KSIZE)) {
X              case ZCAN:
X                   vfile("rzfile: zgethdr returned %d", c);
X                   return ERROR;
X              case ERROR:   /* CRC error */
X                   if ( --n < 0) {
X                        vfile("rzfile: zgethdr returned %d", c);
X                        return ERROR;
X                   }
X                   zmputs(Attn);
X                   continue;
```

```
X             case TIMEOUT:
X                  if ( --n < 0) {
X                       vfile("rzfile: zgethdr returned %d", c);
X                       return ERROR;
X                  }
X                  continue;
X             case GOTCRCW:
X                  n = 10;
X                  putsec(secbuf, Rxcount);
X                  rxbytes += Rxcount;
X                  stohdr(rxbytes);
X                  zshhdr(ZACK, Txhdr);
X                  goto nxthdr;
X             case GOTCRCQ:
X                  n = 10;
X                  putsec(secbuf, Rxcount);
X                  rxbytes += Rxcount;
X                  stohdr(rxbytes);
X                  zshhdr(ZACK, Txhdr);
X                  goto moredata;
X             case GOTCRCG:
X                  n = 10;
X                  putsec(secbuf, Rxcount);
X                  rxbytes += Rxcount;
X                  goto moredata;
X             case GOTCRCE:
X                  n = 10;
X                  putsec(secbuf, Rxcount);
X                  rxbytes += Rxcount;
X                  goto nxthdr;
X             }
X        }
X    }
X}
X
X/*
X * Send a string to the modem, processing for \336 (sleep 1 sec)
X *   and \335 (break signal)
X */
Xzmputs(s)
Xchar *s;
X{
X    register c;
X
X    while (*s) {
X         switch (c = *s++) {
X         case '\336':
X              sleep(1); continue;
X         case '\335':
X              sendbrk(); continue;
X         default:
X              sendline(c);
X         }
```

```
X    }
X}
X
X/*
X * Close the receive dataset, return OK or ERROR
X */
Xcloseit()
X{
X    if (Topipe) {
X         if (pclose(fout)) {
X              return ERROR;
X         }
X         return OK;
X    }
X    if (fclose(fout)==ERROR) {
X         fprintf(stderr, "file close ERROR\n");
X         return ERROR;
X    }
X    if (Modtime) {
X         timep[0] = time(NULL);
X         timep[1] = Modtime;
X         utime(Pathname, timep);
X    }
X    if (Filemode)
X         chmod(Pathname, (07777 & Filemode));
X    return OK;
X}
X
X/*
X * Ack a ZFIN packet, let byegones be byegones
X */
Xackbibi()
X{
X    register n;
X
X    vfile("ackbibi:");
X    Readnum = 1;
X    stohdr(0L);
X    for (n=4; --n>=0; ) {
X         zshhdr(ZFIN, Txhdr);
X         for (;;) {
X              switch (readline(100)) {
X              case 'O':
X                   readline(1);  /* Discard 2nd 'O' */
X                   /* ***** FALL THRU TO ***** */
X              case TIMEOUT:
X                   vfile("ackbibi complete");
X                   return;
X              default:
X                   break;
X              }
X         }
X    }
```

```
X}
X
X/*
X * Local console output simulation
X */
Xbttyout(c)
X{
X    if (Verbose || fromcu())
X         putc(c, stderr);
X}
X
X/*
X * Strip leading ! if present, do shell escape.
X */
Xsys2(s)
Xregister char *s;
X{
X    if (*s == '!')
X         ++s;
X    return system(s);
X}
X/*
X * Strip leading ! if present, do exec.
X */
Xexec2(s)
Xregister char *s;
X{
X    if (*s == '!')
X         ++s;
X    mode(0);
X    execl("/bin/sh", "sh", "-c", s);
X}
X
X#ifdef CASE
X/* CWRU enhancement- create all subdirectories as required
X * Author : jrbammi
X * tested on Unix 4.3BSD ONLY.
X */
X
X#ifdef USG
X#define index strchr
X#endif
X
X/* default directory creation mode, note that this is modified by
X   umask */
X#define DIRMODE 0755
X
X/*
X * Ensure that each subdirectory in the pathname exists.
X * If one doesn'nt, make the rest of the directory
X * Returns ERROR if it has trouble creating a path
X *
X */
```

```
Xint pathensure(name)
Xchar *name;
X{
X    extern char *index();
X
X    if(index(name, '/') == (char *)NULL)
X         /* nothing to check */
X         return OK;
X
X    return pathrest(name, (char *)NULL);
X}
X
X/*
X * check rest of the path recursively
X *
X */
Xint pathrest(name, prev)
Xchar *name, *prev;
X{
X    char previous[256];
X    char component[33];
X    register char *p, *q, *r;
X    register int i;
X    extern char *index();
X
X    if((r = index(name, '/')) == (char *)NULL)
X         /* nothing more to check */
X         return OK;
X
X    /* pick up the next component of the path name, 32 chars max  */
X    for(p = name, q = component, i = 0; (p != r) && (i < 32); i++)
X         *q++ = *p++;
X    *q = '\0';
X
X    if(i >= 32)
X    {
X         log("A component of the path name is longer than 32 characters\n");
X         return ERROR;
X    }
X
X    if(prev == (char *)NULL)
X         strcpy(previous, component);
X    else
X    {
X         strcpy(previous, prev);
X         strcat(previous, "/");
X         strcat(previous, component);
X    }
X
X    if(existd(previous))
X         /* it exists - go do rest */
X         return pathrest(++r, previous);
X
```

```
X    /* does not exist,                    */
X    /* make this component and all its children */
X    return makesubtree(++r, previous);
X}
X
X/*
X * make a subtree
X */
Xint makesubtree(name, prev)
Xchar *name, *prev;
X{
X    char previous[256];
X    char component[33];
X    register char *p, *q, *r;
X    register int i;
X    extern char *index();
X
X    if(mkdir(prev,DIRMODE) != 0)
X    {
X         log("Trouble trying to create subtree\n");
X         return ERROR;
X    }
X    if(Verbose > 2)
X        fprintf(stderr,"Made directory %s\n",prev);
X
X
X    if((r = index(name, '/')) == (char *)NULL)
X         /* nothing more to do */
X         return OK;
X
X    /* pick up the next component of the path name, 32 chars max */
X    for(p = name, q = component, i = 0; (p != r) && (i < 32); i++)
X         *q++ = *p++;
X
X    *q = '\0';
X
X    if(i >= 32)
X    {
X         log("A component of the path name is longer than 32 characters\n");
X         return ERROR;
X    }
X
X    strcpy(previous, prev);
X    strcat(previous, "/");
X    strcat(previous, component);
X
X    /* go do the rest of them */
X    return makesubtree(++r, previous);
X}
X
X
X/*
X * test if a subdirectory exists
```

```
X */
Xint existd(name)
Xregister char *name;
X{
X    struct stat statbuf;
X
X    if(stat(name, &statbuf) != 0)
X        return FALSE;
X
X    return (statbuf.st_mode & S_IFMT) == S_IFDIR;
X
X}
X#endif
SHAR_EOF
chmod 0600 rz.c || echo "restore of rz.c fails"
sed 's/^X//' << 'SHAR_EOF' > sz.1 &&
X'\" Revision Level
X'\" Last Delta     03-10-87
X.TH SZ 1 OMEN
X.SH NAME
Xsz \- XMODEM, YMODEM, ZMODEM Batch file Send
X.SH SYNOPSIS
Xsz
X.RB [\- +1abdefkLlNnopqTtuvy ]
X.I file ...
X.br
Xsz -X
X.RB [\- 1kqtuv ]
X.I file
X.br
Xsz
X.RB [\- 1oqtv ]
X.B "-c COMMAND"
X.br
Xsz
X.RB [\- 1oqtv ]
X.B "-i COMMAND"
X.SH DESCRIPTION
X.B Sz
Xuses the ZMODEM, YMODEM or XMODEM error correcting protocol to send
Xone or more files over a serial port to a variety of programs running under
XPC-DOS, CP/M, Unix, VMS, and other operating systems.
X
X
XThe first form of
X.B sz
Xsends one or more files with ZMODEM or YMODEM batch protocol.
XNormally, only the file name part of the pathname is transmitted.
XOn
X.SM Unix
Xsystems, additional information about the file is transmitted.
XIf the receiving program uses this information,
Xthe transmitted file length controls the exact number of bytes written to
```

```
Xthe output dataset,
Xand the modify time and file mode
Xare set accordingly.
X
XOutput from another program may be piped to
X.B sz
Xfor transmission by specifying the
X.B -1
Xoption and denoting standard input by "-":
X.ce
Xps -ef | sz -
XThe program output is transmitted with the filename sPID.sz
Xwhere PID is the process ID of the
X.B sz
Xprogram.
XIf the environment variable
X.B ONAME
Xis set, that is used instead.
XIn this case, the Unix command:
X.ce
XONAME=con ps -ef|sz -ay -
Xwill send a "file" to the PC-DOS console display.
XThe
X.B -y
Xoption instructs the receiver to open the file for writing unconditionally.
XThe
X.B -a
Xoption
Xcauses the receiver to convert Unix newlines to PC-DOS carriage returns
Xand linefeeds.
X
XUnix
X.B sz
Xsupports
X.B YMODEM-g
Xwith "cbreak" tty mode, XON/XOFF flow control,
Xand the interrupt character set to CAN.
X.B YMODEM-g
X(Professional-YAM
X.B g
Xoption)
Xincreases throughput over error free channels
X(direct connection, X.PC, etc.)
Xby not acknowledging each transmitted sector.
X
X
XThe second form of
X.B sz
Xuses the
X.B -X
Xflag to send a single
X.I file
Xwith
```

```
X.B XMODEM
Xor
X.B XMODEM-1k
Xprotocol.
XThe user must supply the file name to both sending and receiving programs.
X
XIff
X.B sz
Xis invoked with $SHELL set and iff that variable contains the
Xstring
X.I "rsh"
Xor
X.I "rksh"
X(restricted shell), sz operates in restricted mode.
XRestricted mode restricts pathnames to the current directory and
XPUBDIR (usually /usr/spool/uucppublic) and/or subdirectories
Xthereof.
X
X
XThe third form sends a single COMMAND to the receiver for execution.
X.B Sz
Xexits with the COMMAND return value.
XIf COMMAND includes spaces or characters special to the shell,
Xit must be quoted.
X
X
XThe fourth form sends a single COMMAND to the receiver for execution.
X.B Sz
Xexits as soon as the receiver has correctly received the command,
Xbefore it is executed.
X
X
XIf sz is invoked with stdout and stderr to different datasets,
XVerbose is set to 2, causing frame by frame progress reports
Xto stderr.
XThis may be disabled with the
X.B q
Xoption.
X.PP
XThe meanings of the available options are:
X.PP
X.PD 0
X.TP
X.B +
XInstruct the receiver to append transmitted data to an existing file
X(ZMODEM only).
X.TP
X.B 1
XUse file descriptor 1 for ioctls and reads (Unix only).
XBy default, file descriptor 0 is used.
XThis option allows
X.B sz
Xto be used with the
```

```
X.I Professional-YAM
X$
Xcommand.
X.TP
X.B a
XConvert NL characters in the transmitted file to CR/LF.
XThis is done by the sender for XMODEM and YMODEM, by the receiver
Xfor ZMODEM.
X.TP
X.B b
X(ZMODEM) Binary override: transfer file without any translation.
X.TP
X.B "c COMMAND"
XSend COMMAND to the receiver for execution, return with COMMAND\'s exit status.
X.TP
X.B d
XChange all instances of "." to "/" in the transmitted pathname.
XThus, C.omenB0000 (which is unacceptable to MSDOS or CP/M)
Xis transmitted as C/omenB0000.
XIf the resultant filename has more than 8 characters in the stem,
Xa "." is inserted to allow a total of eleven.
X.TP
X.B E
XEscape only Ctrl-X control characters;
Xnormally XON, XOFF, CR-@-CR, and Ctrl-X are escaped.
X.TP
X.B e
XEscape all control characters;
Xnormally XON, XOFF, CR-@-CR, and Ctrl-X are escaped.
X.TP
X.B f
XSend Full pathname.
XNormally directory prefixes are stripped from the transmitted
Xfilename.
X.TP
X.B "i COMMAND"
XSend COMMAND to the receiver for execution, return Immediately
Xupon the receiving program's successful recption of the command.
X.TP
X.B k
X(XMODEM/YMODEM) Send files using 1024 byte blocks
Xrather than the default 128 byte blocks.
X1024 byte packets speed file transfers at high bit rates.
X(ZMODEM streams the data for the best possible throughput.)
X.TP
X.B "L N"
XUse ZMODEM sub-packets of length N.
XA larger N (32 <= N <= 1024) gives slightly higher throughput,
Xa smaller N speeds error recovery.
XThe default is 128 below 300 baud, 256 above 300 baud, or 1024 above 2400 baud.
X.TP
X.B "l N"
XWait for the receiver to acknowledge correct data every
```

```
X.B N
X(32 <= N <= 1024)
Xcharacters.
XThis may be used to avoid network overrun when XOFF flow control is lacking.
X.TP
X.B n
X(ZMODEM) Send each file if
Xdestination file does not exist.
XOverwrite destination file if
Xsource file is newer or longer than the destination file.
X.TP
X.B N
X(ZMODEM) Send each file if
Xdestination file does not exist.
XOverwrite destination file if
Xsource file has different length or date.
X.TP
X.B o
X(ZMODEM) Disable automatic selection of 32 bit CRC.
X.TP
X.B p
X(ZMODEM) Protect existing destination files by skipping transfer if the
Xdestination file exists.
X.TP
X.B q
XQuiet suppresses verbosity.
X.TP
X.B r
XResume interrupted file transfer.
XIf the source file is longer than the destination file,
Xthe transfer commences at the offset in the source file that equals
Xthe length of the destination file.
X.TP
X.B "t tim"
XChange timeout to
X.I tim
Xtenths of seconds.
X.TP
X.B u
XUnlink the file after successful transmission.
X.TP
X.B v
XVerbose
Xcauses a list of file
Xnames to be appended to
X/tmp/szlog .
XMore v's generate more output.
X.TP
X.B X
XSend a single file with
X.B XMODEM
Xor
X.B XMODEM-1k
```

```
Xprotocol.
X.TP
X.B y
XInstruct a ZMODEM receiving program to overwrite any existing file
Xwith the same name.
X.PD
X.SH EXAMPLES
X.ne 7
X.B "ZMODEM File Transfer"
X.br
X.B "$ sz -a *.c"
X.br
XThis single command transfers all .c files in the current Unix directory
Xwith conversion
X.RB ( -a )
Xto end of line conventions appropriate to the receiving environment.
XWith ZMODEM AutoDownload enabled, Professional-YAM  and ZCOMM
Xwill automatically recieve
Xthe files after performing a security check.
X
X.B "ZMODEM Command Download"
X.br
X cpszall:all
X    sz -c "c:;cd /yam/dist"
X    sz -ya $(YD)/*.me
X    sz -yqb y*.exe
X    sz -c "cd /yam"
X    sz -i "!insms"
X.br
XThis Makefile fragment uses
X.B sz
Xto issue commands to Professional-YAM to change current disk and directory.
XNext,
X.B sz
Xtransfers the
X.I .me
Xfiles from the $YD directory, commanding the receiver to overwrite the old files
Xand to convert from Unix end of line conventions to PC-DOS conventions.
XThe third line transfers some
X.I .exe
Xfiles.
XThe fourth and fifth lines command Pro-YAM to
Xchange directory and execute a PC-DOS batch file
X.I insms .
XSince the batch file takes considerable time, the
X.B "-i"
Xform is used to allow
X.B sz
Xto exit immediately.
X
X.B "XMODEM File Transfer"
X.br
X$
```

```
X.B "sz -Xa foo.c"
X.br
X.B "ESC"
X.br
X.B "rx foo.c"
X.br
XThe above three commands transfer a single file
Xfrom Unix to a PC and Crosstalk XVI 3.6,
Xtranslating Unix newlines to DOS CR/LF.
X
X32 bit CRC code courtesy Gary S. Brown.
X.SH SEE ALSO
Xrz(omen),
XZMODEM.DOC,
XYMODEM.DOC,
XProfessional-YAM,
XIMP(CP/M),
Xsq(omen),
Xtodos(omen),
Xtocpm(omen),
Xtomac(omen),
Xyam(omen)
X
XCompile time options required for various operating systems are described in
Xthe source file.
X.SH "VMS VERSION"
XThe VMS version does not transmit the file date.
XThe VMS version calculates the file length by reading the file
Xand counting the bytes.
X
XThe VMS version does not support YMODEM-g or ZMODEM.
X
XWhen VMS is lightly loaded, the response time may be too quick for MODEM7
Xunless the MODEM7
X.B "q"
Xmodifier is used.
X
XThe VMS C standard i/o package and RMS sometimes interact to modify
Xfile contents unexpectedly.
X.SH FILES
Xsz.c, rbsb.c, zm.c, zmodem.h source files
X
X/tmp/szlog stores debugging output (sz -vv)
X.SH "TESTING FEATURE"
XThe command "sz -T file"
Xexercises the
X.B Attn
Xsequence error recovery by commanding
Xerrors with unterminated packets.
XThe receiving program should complain five times about
Xbinary data packets being too long.
XEach time
X.B sz
```

```
Xis interrupted,
Xit should send a ZDATA header followed by another defective packet.
XIf the receiver does not detect five long data packets,
Xthe
X.B Attn
Xsequence is not interrupting the sender, and the
X.B Myattn
Xstring in
X.B sz.c
Xmust be modified.
X
XAfter 5 packets,
X.B sz
Xstops the "transfer" and
Xprints the total number of characters "sent" (Tcount).
XThe difference between Tcount and 5120 represents the number of characters
Xstored in various buffers when the Attn sequence is generated.
X.SH BUGS
XXMODEM transfers add up to 127 garbage bytes per file (1023 bytes with
XXMODEM-k).
XMost YMODEM programs use the file length transmitted at the beginning of the
Xtransfer to prune the file to the correct length; this may cause problems with
Xsource files that grow during the course of the transfer.
XThis problem does not pertain to ZMODEM transfers, which preserve the exact
Xfile length unconditionally.
X
XMost ZMODEM options are merely passed to the receiving program;
Xsome do not implement all these options.
X
XCircular buffering and a ZMODEM sliding window should be used
Xwhen input is from pipes instead of acknowledging frames each 1024 bytes.
XIf no files can be opened,
X.B sz
Xsends a ZMODEM command to echo a suitable complaint;
Xperhaps it should check for the presence of at least one accessible file before
Xgetting hot and bothered.
XThe test mode leaves a zero length file on the receiving system.
X
XSome high speed modems have a firmware bug that drops characters when the
Xdirection of high speed transmisson is reversed.
XThe environment variable ZNULLS may be used to specify the number of nulls to
Xsend before a ZDATA frame.
XValues of 101 for a 4.77 mHz PC and 124 for an AT are typical.
X.SH CWRU `ENHANCEMENTS'
XIf you compiled
X.B sz
Xwith the preprocessor symbol `CASE' defined then you will be able to use
Xthe following `features'.
X.sp 1
XIf sz if given the name of a directory as its arguement, then it will
Xsend the contents of the named directory, and of all the subdirectories.
XIf you need to send full path names, you still have to specify the
X.I -f
```

```
Xoption to
X.I sz.
X.sp 1
XIf you want to skip a branch of a directory, and do the rest, then you
Xmay use the
X.I -P name
Xoption, to `prune' off that branch of the tree. You may specify multiple
X.I -P name
Xoptions on the command line. They may appear anywhere on the command line,
Xso long as they appear before the name of a branch they prune, if thats
Xat the top level.
X.sp 1
XFor instance, let us suppose that you want to back up your unix system
Xto another system over the serial line, using zmodem. You  want to
Xback up all directories, except /tmp and say /usr/lib. Then you would
Xissue the following command:
X.sp 1
X.ce
Xsz -f -P /tmp -P /usr/lib /
X.sp 1
Xand go away for a looooong time!
X.sp 1
XThe CWRU `enhancements' use Berkeley Unix specific code, and have been
Xtested only under BSD4.3. When making
X.I sz
Xplease make sure that you are using the appropriate flavor of Unix, before
Xdefining the pre processor symbol `CASE'.
SHAR_EOF
chmod 0600 sz.1 || echo "restore of sz.1 fails"
sed 's/^X//' << 'SHAR_EOF' > sz.c &&
X#define VERSION "sz 1.26 03-10-87"
X#define PUBDIR "/usr/spool/uucppublic"
X
X/*% cc -O -K -i -DNFGVMIN -DREADCHECK sz.c -lx -o sz; size sz
X *  (above for SYS III/V Xenix)
X
X * sz.c By Chuck Forsberg
X *
X *  cc -O sz.c -o sz       USG (SYS III/V) Unix
X *  cc -O -DSVR2 sz.c -o sz     Sys V Release 2 with non-blocking input
X *                      Define to allow reverse channel checking
X *  cc -O -DV7  sz.c -o sz      Unix Version 7, 2.8 - 4.3 BSD
X *
X *  cc -O -K -i -DNFGVMIN -DREADCHECK sz.c -lx -o sz   Xenix
X *
X *  ln sz sb           **** All versions ****
X *
X *
X *  ******* Some systems (Venix, Coherent, Regulus) do not *******
X *  ******* support tty raw mode read(2) identically to    *******
X *  ******* Unix. ONEREAD must be defined to force one     *******
X *  ******* character reads for these systems.            *******
X *
```

```
X * A program for Unix to send files and commands to computers running
X *  Professional-YAM, PowerCom, YAM, IMP, or programs supporting Y/XMODEM.
X *
X *  Sz uses buffered I/O to greatly reduce CPU time compared to UMODEM.
X *
X *  USG UNIX (3.0) ioctl conventions courtesy Jeff Martin
X */
X
X
Xchar *substr(), *getenv();
X
X#define LOGFILE "/tmp/szlog"
X#define zperr vfile
X
X#include <stdio.h>
X#include <signal.h>
X#include <setjmp.h>
X#include <ctype.h>
X
X#define PATHLEN 256
X#define OK 0
X#define FALSE 0
X#define TRUE 1
X#define ERROR (-1)
X
X#define HOWMANY 2
Xint Zmodem=0;          /* ZMODEM protocol requested */
Xunsigned Baudrate;
Xint Fromcu = 0;        /* Were called from cu or yam */
X#include "rbsb.c" /* most of the system dependent stuff here */
X
X/*
X * Attention string to be executed by receiver to interrupt streaming data
X *  when an error is detected.  A pause (0336) may be needed before the
X *  ^C (03) or after it.
X */
X#ifdef READCHECK
Xchar Myattn[] = { 0 };
X#else
X#ifdef USG
Xchar Myattn[] = { 03, 0336, 0 };
X#else
Xchar Myattn[] = { 0 };
X#endif
X#endif
X
XFILE *in;
X
X/* Ward Christensen / CP/M parameters - Don't change these! */
X#define ENQ 005
X#define CAN ('X'&037)
X#define XOFF ('s'&037)
X#define XON ('q'&037)
```

```
X#define SOH 1
X#define STX 2
X#define EOT 4
X#define ACK 6
X#define NAK 025
X#define CPMEOF 032
X#define WANTCRC 0103  /* send C not NAK to get crc not checksum */
X#define WANTG 0107    /* Send G not NAK to get nonstop batch xmsn */
X#define TIMEOUT (-2)
X#define RCDO (-3)
X#define RETRYMAX 10
X#define SECSIZ 128    /* cp/m's Magic Number record size */
X#define KSIZE 1024
X
Xchar Lastrx;
Xchar Crcflg;
Xint Wcsmask=0377;
Xint Verbose=0;
Xint Modem=0;      /* MODEM - don't send pathnames */
Xint Restricted=0; /* restricted; no /.. or ../ in filenames */
Xint Quiet=0;      /* overrides logic that would otherwise set verbose */
Xint Ascii=0;      /* Add CR's for brain damaged programs */
Xint Fullname=0;       /* transmit full pathname */
Xint Unlinkafter=0;    /* Unlink file after it is sent */
Xint Dottoslash=0; /* Change foo.bar.baz to foo/bar/baz */
Xint firstsec;
Xint errcnt=0;         /* number of files unreadable */
Xint blklen=SECSIZ;         /* length of transmitted records */
Xint Optiong;      /* Let it rip no wait for sector ACK's */
Xint Noeofseen;
Xint Totsecs;      /* total number of sectors this file */
Xchar txbuf[KSIZE];
Xint Filcnt=0;         /* count of number of files opened */
Xint Lfseen=0;
Xunsigned Rxbuflen = 16384; /* Receiver's max buffer length */
Xint Tframlen = 0; /* Override for tx frame length */
Xint blkopt=0;         /* Override value for zmodem blklen */
Xint Rxflags = 0;
Xint Wantfcs32 = TRUE;  /* want to send 32 bit FCS */
SHAR_EOF
echo "End of part 3, continue with part 4"
echo "4" > s2_seq_.tmp
exit 0


From 5719327395613068808
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,48b5679a4fa212ee,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-27 11:42:39 PST
Path: utzoo!attcan!uunet!seismo!sundc!pitstop!sun!decwrl!ucbvax!agate!saturn!ssyx!
koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
```

Newsgroups: comp.sources.atari.st
Subject: v01i039: rzsz -- UNIX side for zmdm file transfer program part04/05
Keywords: UNIX
Message-ID: <3486@saturn.ucsc.edu>
Date: 27 May 88 18:42:39 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1426
Approved: koreth@ssyx.ucsc.edu

Submitted-by: bammi@mandrill.ces.cwru.edu (Jwahar R. Bammi)
Posting-number: Volume 1, Issue 39
Archive-name: rzsz/part04

```
#!/bin/sh
# this is part 4 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file sz.c continued
#
CurArch=4
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> sz.c
Xchar Lzconv; /* Local ZMODEM file conversion request */
Xchar Lzmanag;     /* Local ZMODEM file management request */
Xchar Lztrans;
Xchar zconv;       /* ZMODEM file conversion request */
Xchar zmanag;      /* ZMODEM file management request */
Xchar ztrans;      /* ZMODEM file transport request */
Xint Command;      /* Send a command, then exit. */
Xchar *Cmdstr;          /* Pointer to the command string */
Xint Cmdtries = 11;
Xint Cmdack1;      /* Rx ACKs command, then do it */
Xint Exitcode;
Xint Testattn;          /* Force receiver to send Attn, etc with qbf. */
Xchar *qbf="The quick brown fox jumped over the lazy dog's back 1234567890\r\n";
Xlong Lastread;         /* Beginning offset of last buffer read */
Xint Lastc;        /* Count of last buffer read or -1 */
Xint Dontread;          /* Don't read the buffer, it's still there */
X
Xjmp_buf tohere;        /* For the interrupt on RX timeout */
Xjmp_buf intrjmp;  /* For the interrupt on RX CAN */
X
X/* called by signal interrupt or terminate to clean things up */
Xbibi(n)
X{
X    canit(); fflush(stdout); mode(0);
```

```
X    fprintf(stderr, "sz: caught signal %d; exiting\n", n);
X    if (n == SIGQUIT)
X         abort();
X    exit(128+n);
X}
X/* Called when Zmodem gets an interrupt (^X) */
Xonintr()
X{
X    signal(SIGINT, SIG_IGN);
X    longjmp(intrjmp, -1);
X}
X
X
X#define sendline(c) putchar(c & Wcsmask)
X
X#define xsendline(c) putchar(c)
X
Xflushmo()
X{
X    fflush(stdout);
X}
X
X#define ZKER
Xint Zctlesc; /* Encode control characters */
Xint Nozmodem = 0; /* If invoked as "sb" */
Xchar *Progname = "sz";
X#include "zm.c"
X
X
X#ifdef CASE
XMAIN(argc, argv)
X#else
Xmain(argc, argv)
X#endif
Xint argc;
Xchar **argv;
X{
X    register char *cp;
X    register npats;
X    int agcnt; char **agcv;
X    char **patts;
X    static char xXbuf[BUFSIZ];
X
X    if ((cp = getenv("ZNULLS")) && *cp)
X         Znulls = atoi(cp);
X    if ((cp=getenv("SHELL")) && (substr(cp, "rsh") || substr(cp, "rksh")))
X         Restricted=TRUE;
X    chkinvok(argv[0]);
X
X    Rxtimeout = 600;
X    npats=0;
X    if (argc<2)
X         usage();
```

```
X    setbuf(stdout, xXbuf);
X    while (--argc) {
X         cp = *++argv;
X         if (*cp++ == '-' && *cp) {
X              while ( *cp) {
X                   switch(*cp++) {
X                   case '+':
X                        Lzmanag = ZMAPND; break;
X                   case '1':
X                        iofd = 1; break;
X#ifdef CSTOPB
X                   case '2':
X                        Twostop = TRUE; break;
X#endif
X                   case '7':
X                        Wcsmask=0177; break;
X                   case 'a':
X                        Lzconv = ZCNL;
X                        Ascii = TRUE; break;
X                   case 'b':
X                        Lzconv = ZCBIN; break;
X                   case 'C':
X                        if (--argc < 1) {
X                             usage();
X                        }
X                        Cmdtries = atoi(*++argv);
X                        break;
X                   case 'i':
X                        Cmdack1 = ZCACK1;
X                        /* **** FALL THROUGH TO **** */
X                   case 'c':
X                        if (--argc != 1) {
X                             usage();
X                        }
X                        Command = TRUE;
X                        Cmdstr = *++argv;
X                        break;
X                   case 'd':
X                        ++Dottoslash;
X                        /* **** FALL THROUGH TO **** */
X                   case 'f':
X                        Fullname=TRUE; break;
X                   case 'E':
X                        Zctlesc = -1; break;
X                   case 'e':
X                        Zctlesc = 1; break;
X                   case 'k':
X                        blklen=KSIZE; break;
X                   case 'L':
X                        if (--argc < 1) {
X                             usage();
X                        }
X                        blkopt = atoi(*++argv);
```

```
X                    if (blkopt<24 || blkopt>1024)
X                         usage();
X                    break;
X               case 'l':
X                    if (--argc < 1) {
X                         usage();
X                    }
X                    Tframlen = atoi(*++argv);
X                    if (Tframlen<32 || Tframlen>1024)
X                         usage();
X                    break;
X               case 'N':
X                    Lzmanag = ZMDIFF;  break;
X               case 'n':
X                    Lzmanag = ZMNEW;  break;
X               case 'o':
X                    Wantfcs32 = FALSE; break;
X               case 'p':
X                    Lzmanag = ZMPROT;  break;
X               case 'r':
X                    Lzconv = ZCRESUM;
X               case 'q':
X                    Quiet=TRUE; Verbose=0; break;
X               case 't':
X                    if (--argc < 1) {
X                         usage();
X                    }
X                    Rxtimeout = atoi(*++argv);
X                    if (Rxtimeout<10 || Rxtimeout>1000)
X                         usage();
X                    break;
X               case 'T':
X                    Testattn = TRUE; break;
X               case 'u':
X                    ++Unlinkafter; break;
X               case 'v':
X                    ++Verbose; break;
X               case 'X':
X                    ++Modem; break;
X               case 'y':
X                    Lzmanag = ZMCLOB; break;
X               default:
X                    usage();
X               }
X          }
X     }
X     else if ( !npats && argc>0) {
X          if (argv[0][0]) {
X               npats=argc;
X               patts=argv;
X               if ( !strcmp(*patts, "-"))
X                    iofd = 1;
X          }
```

```
X         }
X    }
X    if (npats < 1 && !Command)
X         usage();
X    if (Verbose) {
X         if (freopen(LOGFILE, "a", stderr)==NULL) {
X              printf("Can't open log file %s\n",LOGFILE);
X              exit(0200);
X         }
X         setbuf(stderr, NULL);
X    }
X    if ((Fromcu=from_cu()) && !Quiet) {
X         if (Verbose == 0)
X              Verbose = 2;
X    }
X
X    mode(1);
X
X    if (signal(SIGINT, bibi) == SIG_IGN) {
X         signal(SIGINT, SIG_IGN); signal(SIGKILL, SIG_IGN);
X    } else {
X         signal(SIGINT, bibi); signal(SIGKILL, bibi);
X    }
X    if ( !Fromcu)
X         signal(SIGQUIT, SIG_IGN);
X
X    if ( !Modem) {
X         if (!Nozmodem) {
X              printf("rz\r");  fflush(stdout);
X         }
X         if (!Command && !Quiet && Verbose != 1) {
X              fprintf(stderr, "%s: %d file%s requested:\r\n",
X               Progname, npats, npats>1?"s":"");
X              for ( agcnt=npats, agcv=patts; --agcnt>=0; ) {
X                   fprintf(stderr, "%s ", *agcv++);
X              }
X              fprintf(stderr, "\r\n");
X              printf("\r\n\bSending in Batch Mode\r\n");
X         }
X         if (!Nozmodem) {
X              stohdr(0L);
X              if (Command)
X                   Txhdr[ZF0] = ZCOMMAND;
X              zshhdr(ZRQINIT, Txhdr);
X         }
X    }
X    fflush(stdout);
X
X    if (Command) {
X         if (getzrxinit()) {
X              Exitcode=0200; canit();
X         }
X         else if (zsendcmd(Cmdstr, 1+strlen(Cmdstr))) {
```

```
X             Exitcode=0200; canit();
X         }
X    } else if (wcsend(npats, patts)==ERROR) {
X         Exitcode=0200;
X         canit();
X    }
X    fflush(stdout);
X    mode(0);
X    exit((errcnt != 0) | Exitcode);
X    /*NOTREACHED*/
X}
X
Xwcsend(argc, argp)
Xchar *argp[];
X{
X    register n;
X
X    Crcflg=FALSE;
X    firstsec=TRUE;
X    for (n=0; n<argc; ++n) {
X         Totsecs = 0;
X         if (wcs(argp[n])==ERROR)
X              return ERROR;
X    }
X    Totsecs = 0;
X    if (Filcnt==0) {   /* bitch if we couldn't open ANY files */
X         if (1) {
X              Command = TRUE;
X              Cmdstr = "echo \"sz: Can't open any requested files\"";
X              if (getnak()) {
X                   Exitcode=0200; canit();
X              }
X              if (!Zmodem)
X                   canit();
X              else if (zsendcmd(Cmdstr, 1+strlen(Cmdstr))) {
X                   Exitcode=0200; canit();
X              }
X              Exitcode = 1; return OK;
X         }
X         canit();
X         fprintf(stderr,"\r\nCan't open any requested files.\r\n");
X         return ERROR;
X    }
X    if (Zmodem)
X         saybibi();
X    else
X         wctxpn("");
X    return OK;
X}
X
Xwcs(oname)
Xchar *oname;
X{
```

```
X    register c;
X    register char *p;
X    struct stat f;
X    char name[PATHLEN];
X
X    strcpy(name, oname);
X
X    if (Restricted) {
X         /* restrict pathnames to current tree or uucppublic */
X         if ( substr(name, "../")
X          || (name[0]== '/' && strncmp(name, PUBDIR, strlen(PUBDIR))) ) {
X              canit();
X              fprintf(stderr,"\r\nsz:\tSecurity Violation\r\n");
X              return ERROR;
X         }
X    }
X
X    if ( !strcmp(oname, "-")) {
X         if ((p = getenv("ONAME")) && *p)
X              strcpy(name, p);
X         else
X              sprintf(name, "s%d.sz", getpid());
X         in = stdin;
X    }
X    else if ((in=fopen(oname, "r"))==NULL) {
X         ++errcnt;
X         return OK;    /* pass over it, there may be others */
X    }
X    ++Noeofseen;  Lastread = 0;  Lastc = -1; Dontread = FALSE;
X    /* Check for directory or block special files */
X    fstat(fileno(in), &f);
X    c = f.st_mode & S_IFMT;
X    if (c == S_IFDIR || c == S_IFBLK) {
X         fclose(in);
X         return OK;
X    }
X
X    ++Filcnt;
X    switch (wctxpn(name)) {
X    case ERROR:
X         return ERROR;
X    case ZSKIP:
X         return OK;
X    }
X    if (!Zmodem && wctx()==ERROR)
X         return ERROR;
X    if (Unlinkafter)
X         unlink(oname);
X    return 0;
X}
X
X/*
X * generate and transmit pathname block consisting of
```

```
X *  pathname (null terminated),
X *  file length, mode time and file mode in octal
X *  as provided by the Unix fstat call.
X *  N.B.: modifies the passed name, may extend it!
X */
Xwctxpn(name)
Xchar *name;
X{
X    register char *p, *q;
X    char name2[PATHLEN];
X    struct stat f;
X
X    if (Modem) {
X#ifdef CASE
X        if (Verbose && (in!=stdin) && *name && fstat(fileno(in), &f)!= -1) {
X#else
X        if ((in!=stdin) && *name && fstat(fileno(in), &f)!= -1) {
X#endif
X             fprintf(stderr, "Sending %s, %ld blocks: ",
X               name, f.st_size>>7);
X        }
X#ifdef CASE
X        if(Verbose)
X#endif
X            fprintf(stderr, "Give your local XMODEM receive command now.\r\n");
X        return OK;
X    }
X    logent("\r\nAwaiting pathname nak for %s\r\n", *name?name:"<END>");
X    if ( !Zmodem)
X        if (getnak())
X             return ERROR;
X
X    q = (char *) 0;
X    if (Dottoslash) {      /* change . to . */
X        for (p=name; *p; ++p) {
X             if (*p == '/')
X                  q = p;
X             else if (*p == '.')
X                  *(q=p) = '/';
X        }
X        if (q && strlen(++q) > 8) { /* If name>8 chars */
X             q += 8;            /*   make it .ext */
X             strcpy(name2, q);  /* save excess of name */
X             *q = '.';
X             strcpy(++q, name2);    /* add it back */
X        }
X    }
X
X    for (p=name, q=txbuf ; *p; )
X        if ((*q++ = *p++) == '/' && !Fullname)
X             q = txbuf;
X    *q++ = 0;
X    p=q;
```

```
X    while (q < (txbuf + KSIZE))
X         *q++ = 0;
X    if (!Ascii && (in!=stdin) && *name && fstat(fileno(in), &f)!= -1)
X         sprintf(p, "%lu %lo %o", f.st_size, f.st_mtime, f.st_mode);
X    /* force 1k blocks if name won't fit in 128 byte block */
X    if (txbuf[125])
X         blklen=KSIZE;
X    else {        /* A little goodie for IMP/KMD */
X         if (Zmodem)
X              blklen = SECSIZ;
X         txbuf[127] = (f.st_size + 127) >>7;
X         txbuf[126] = (f.st_size + 127) >>15;
X    }
X    if (Zmodem)
X         return zsendfile(txbuf, 1+strlen(p)+(p-txbuf));
X    if (wcputsec(txbuf, 0, SECSIZ)==ERROR)
X         return ERROR;
X    return OK;
X}
X
Xgetnak()
X{
X    register firstch;
X
X    Lastrx = 0;
X    for (;;) {
X         switch (firstch = readock(800,1)) {
X         case ZPAD:
X              if (getzrxinit())
X                   return ERROR;
X              Ascii = 0;
X              return FALSE;
X         case TIMEOUT:
X              logent("Timeout on pathname\n");
X              return TRUE;
X         case WANTG:
X#ifdef USG
X              mode(2); /* Set cbreak, XON/XOFF, etc. */
X#endif
X              Optiong = TRUE;
X              blklen=KSIZE;
X         case WANTCRC:
X              Crcflg = TRUE;
X         case NAK:
X              return FALSE;
X         case CAN:
X              if ((firstch = readock(20,1)) == CAN && Lastrx == CAN)
X                   return TRUE;
X         default:
X              break;
X         }
X         Lastrx = firstch;
X    }
```

```
X}
X
X
Xwctx()
X{
X    register int sectnum, attempts, firstch;
X
X    firstsec=TRUE;
X
X    while ((firstch=readock(Rxtimeout, 2))!=NAK && firstch != WANTCRC
X      && firstch != WANTG && firstch!=TIMEOUT && firstch!=CAN)
X        ;
X    if (firstch==CAN) {
X         logent("Receiver CANcelled\n");
X         return ERROR;
X    }
X    if (firstch==WANTCRC)
X         Crcflg=TRUE;
X    if (firstch==WANTG)
X         Crcflg=TRUE;
X    sectnum=1;
X    while (filbuf(txbuf, blklen)) {
X         if (wcputsec(txbuf, sectnum, blklen)==ERROR) {
X              return ERROR;
X         } else
X              sectnum++;
X    }
X    if (Verbose>1)
X         fprintf(stderr, " Closing ");
X    fclose(in);
X    attempts=0;
X    do {
X         logent(" EOT ");
X         purgeline();
X         sendline(EOT);
X         fflush(stdout);
X         ++attempts;
X    }
X         while ((firstch=(readock(Rxtimeout, 1)) != ACK) && attempts < RETRYMAX);
X    if (attempts == RETRYMAX) {
X         logent("No ACK on EOT\n");
X         return ERROR;
X    }
X    else
X         return OK;
X}
X
Xwcputsec(buf, sectnum, cseclen)
Xchar *buf;
Xint sectnum;
Xint cseclen; /* data length of this sector to send */
X{
X    register checksum, wcj;
```

```
X    register char *cp;
X    unsigned oldcrc;
X    int firstch;
X    int attempts;
X
X    firstch=0;    /* part of logic to detect CAN CAN */
X
X    if (Verbose>1)
X         fprintf(stderr, "\rSector %3d %2dk ", Totsecs, Totsecs/8 );
X    for (attempts=0; attempts <= RETRYMAX; attempts++) {
X         Lastrx= firstch;
X         sendline(cseclen==KSIZE?STX:SOH);
X         sendline(sectnum);
X         sendline(-sectnum -1);
X         oldcrc=checksum=0;
X         for (wcj=cseclen,cp=buf; --wcj>=0; ) {
X              sendline(*cp);
X              oldcrc=updcrc((0377& *cp), oldcrc);
X              checksum += *cp++;
X         }
X         if (Crcflg) {
X              oldcrc=updcrc(0,updcrc(0,oldcrc));
X              sendline((int)oldcrc>>8);
X              sendline((int)oldcrc);
X         }
X         else
X              sendline(checksum);
X
X         if (Optiong) {
X              firstsec = FALSE; return OK;
X         }
X         firstch = readock(Rxtimeout, (Noeofseen&&sectnum) ? 2:1);
Xgotnak:
X         switch (firstch) {
X         case CAN:
X              if(Lastrx == CAN) {
Xcancan:
X                   logent("Cancelled\n");  return ERROR;
X              }
X              break;
X         case TIMEOUT:
X              logent("Timeout on sector ACK\n"); continue;
X         case WANTCRC:
X              if (firstsec)
X                   Crcflg = TRUE;
X         case NAK:
X              logent("NAK on sector\n"); continue;
X         case ACK:
X              firstsec=FALSE;
X              Totsecs += (cseclen>>7);
X              return OK;
X         case ERROR:
X              logent("Got burst for sector ACK\n"); break;
```

```
X         default:
X              logent("Got %02x for sector ACK\n", firstch); break;
X         }
X         for (;;) {
X              Lastrx = firstch;
X              if ((firstch = readock(Rxtimeout, 2)) == TIMEOUT)
X                   break;
X              if (firstch == NAK || firstch == WANTCRC)
X                   goto gotnak;
X              if (firstch == CAN && Lastrx == CAN)
X                   goto cancan;
X         }
X    }
X    logent("Retry Count Exceeded\n");
X    return ERROR;
X}
X
X/* fill buf with count chars padding with ^Z for CPM */
Xfilbuf(buf, count)
Xregister char *buf;
X{
X    register c, m;
X
X    if ( !Ascii) {
X         m = read(fileno(in), buf, count);
X         if (m <= 0)
X              return 0;
X         while (m < count)
X              buf[m++] = 032;
X         return count;
X    }
X    m=count;
X    if (Lfseen) {
X         *buf++ = 012; --m; Lfseen = 0;
X    }
X    while ((c=getc(in))!=EOF) {
X         if (c == 012) {
X              *buf++ = 015;
X              if (--m == 0) {
X                   Lfseen = TRUE; break;
X              }
X         }
X         *buf++ =c;
X         if (--m == 0)
X              break;
X    }
X    if (m==count)
X         return 0;
X    else
X         while (--m>=0)
X              *buf++ = CPMEOF;
X    return count;
X}
```

```
X/* fill buf with count chars */
Xzfilbuf(buf, count)
Xregister char *buf;
X{
X    register c, m;
X
X    m=count;
X    while ((c=getc(in))!=EOF) {
X         *buf++ =c;
X         if (--m == 0)
X              break;
X    }
X    return (count - m);
X}
X
X/* VARARGS1 */
Xvfile(f, a, b, c)
Xregister char *f;
X{
X    if (Verbose > 1) {
X         fprintf(stderr, f, a, b, c);
X         fprintf(stderr, "\n");
X    }
X}
X
X
Xalrm()
X{
X    longjmp(tohere, -1);
X}
X
X
X/*
X * readock(timeout, count) reads character(s) from file descriptor 0
X *  (1 <= count <= 3)
X * it attempts to read count characters. If it gets more than one,
X * it is an error unless all are CAN
X * (otherwise, only normal response is ACK, CAN, or C)
X *  Only looks for one if Optiong, which signifies cbreak, not raw input
X *
X * timeout is in tenths of seconds
X */
Xreadock(timeout, count)
X{
X    register int c;
X    static char byt[5];
X
X    if (Optiong)
X         count = 1;    /* Special hack for cbreak */
X
X    fflush(stdout);
X    if (setjmp(tohere)) {
X         logent("TIMEOUT\n");
```

```
X         return TIMEOUT;
X    }
X    c = timeout/10;
X    if (c<2)
X         c=2;
X    if (Verbose>3) {
X         fprintf(stderr, "Timeout=%d Calling alarm(%d) ", timeout, c);
X         byt[1] = 0;
X    }
X    signal(SIGALRM, alrm); alarm(c);
X#ifdef ONEREAD
X    c=read(iofd, byt, 1);       /* regulus raw read is unique */
X#else
X    c=read(iofd, byt, count);
X#endif
X    alarm(0);
X    if (Verbose>5)
X         fprintf(stderr, "ret cnt=%d %x %x\n", c, byt[0], byt[1]);
X    if (c<1)
X         return TIMEOUT;
X    if (c==1)
X         return (byt[0]&0377);
X    else
X         while (c)
X              if (byt[--c] != CAN)
X                   return ERROR;
X    return CAN;
X}
Xreadline(n)
X{
X    return (readock(n, 1));
X}
X
X
Xpurgeline()
X{
X#ifdef USG
X    ioctl(iofd, TCFLSH, 0);
X#else
X    lseek(iofd, 0L, 2);
X#endif
X}
X
X
X/* send cancel string to get the other end to shut up */
Xcanit()
X{
X    static char canistr[] = {
X     24,24,24,24,24,24,24,24,24,8,8,8,8,8,8,8,8,8,0
X    };
X
X    printf(canistr);
X    fflush(stdout);
```

```
X}
X
X/*VARARGS1*/
Xlogent(a, b, c)
Xchar *a, *b, *c;
X{
X    if(Verbose)
X         fprintf(stderr, a, b, c);
X}
X
X/*
X * return 1 iff stdout and stderr are different devices
X *  indicating this program operating with a modem on a
X *  different line
X */
Xfrom_cu()
X{
X    struct stat a, b;
X    fstat(1, &a); fstat(2, &b);
X    return (a.st_rdev != b.st_rdev);
X}
X
X/*
X * substr(string, token) searches for token in string s
X * returns pointer to token within string if found, NULL otherwise
X */
Xchar *
Xsubstr(s, t)
Xregister char *s,*t;
X{
X    register char *ss,*tt;
X    /* search for first char of token */
X    for (ss=s; *s; s++)
X         if (*s == *t)
X              /* compare token with substring */
X              for (ss=s,tt=t; ;) {
X                   if (*tt == 0)
X                        return s;
X                   if (*ss++ != *tt++)
X                        break;
X              }
X    return NULL;
X}
X
Xchar *babble[] = {
X    "Send file(s) with ZMODEM/YMODEM/XMODEM Protocol",
X    "    (Y) = Option applies to YMODEM only",
X    "    (Z) = Option applies to ZMODEM only",
X    "Usage:  sz [-12+aBbdefkLlNnquvXy] [-] file ...",
X    "    sz [-1Beqv] -c COMMAND",
X    "    1 Use stdout for modem input",
X#ifdef CSTOPB
X    "    2 Use 2 stop bits",
```

```
X#endif
X    "    + Append to existing destination file (Z)",
X    "    a (ASCII) change NL to CR/LF",
X    "    b Binary file transfer override",
X    "    c send COMMAND (Z)",
X    "    d Change '.' to '/' in pathnames (Y/Z)",
X    "    e Escape all control characters (Z)",
X    "    f send Full pathname (Y/Z)",
X    "    i send COMMAND, ack Immediately (Z)",
X    "    k Send 1024 byte packets (Y)",
X    "    L N Limit subpacket length to N bytes (Z)",
X    "    l N Limit frame length to N bytes (l>=L) (Z)",
X    "    n send file if source Newer or longer (Z)",
X    "    N send file if source different length or date (Z)",
X    "    o Use 16 bit CRC instead of 32 bit CRC (Z)",
X    "    p Protect existing destination file (Z)",
X    "    r Resume/Recover interrupted file transfer (Z)",
X    "    q Quiet (no progress reports)",
X    "    u Unlink file after transmission",
X    "    v Verbose - debugging information",
X    "    X XMODEM protocol - send no pathnames",
X    "    y Yes, overwrite existing file (Z)",
X    "- as pathname sends standard input as sPID.sz or environment ONAME",
X    ""
X};
X
Xusage()
X{
X    char **pp;
X
X    for (pp=babble; **pp; ++pp)
X        fprintf(stderr, "%s\n", *pp);
X    fprintf(stderr, "%s for %s by Chuck Forsberg  ", VERSION, OS);
X    exit(1);
X}
X
X/*
X * Get the receiver's init parameters
X */
Xgetzrxinit()
X{
X    register n;
X    struct stat f;
X
X    for (n=10; --n>=0; ) {
X
X        switch (zgethdr(Rxhdr, 1)) {
X        case ZCHALLENGE:   /* Echo receiver's challenge numbr */
X            stohdr(Rxpos);
X            zshhdr(ZACK, Txhdr);
X            continue;
X        case ZCOMMAND:         /* They didn't see out ZRQINIT */
X            stohdr(0L);
```

```
X             zshhdr(ZRQINIT, Txhdr);
X             continue;
X         case ZRINIT:
X             Rxflags = 0377 & Rxhdr[ZF0];
X             Txfcs32 = (Wantfcs32 && (Rxflags & CANFC32));
X             Rxbuflen = (0377 & Rxhdr[ZP0])+((0377 & Rxhdr[ZP1])<<8);
X             vfile("Rxbuflen=%d Tframlen=%d", Rxbuflen, Tframlen);
X             if ( !Fromcu)
X                  signal(SIGINT, SIG_IGN);
X#ifdef USG
X             mode(2);  /* Set cbreak, XON/XOFF, etc. */
X#endif
X#ifndef READCHECK
X#ifndef USG
X             /* Use 1024 byte frames if no sample/interrupt */
X             if (Rxbuflen < 32 || Rxbuflen > 1024) {
X                  Rxbuflen = 1024;
X                  vfile("Rxbuflen=%d", Rxbuflen);
X             }
X#endif
X#endif
X             /* Override to force shorter frame length */
X             if (Rxbuflen && (Rxbuflen>Tframlen) && (Tframlen>=32))
X                  Rxbuflen = Tframlen;
X             if ( !Rxbuflen && (Tframlen>=32) && (Tframlen<=1024))
X                  Rxbuflen = Tframlen;
X             vfile("Rxbuflen=%d", Rxbuflen);
X
X             /* If using a pipe for testing set lower buf len */
X             fstat(iofd, &f);
X             if ((f.st_mode & S_IFMT) != S_IFCHR
X               && (Rxbuflen == 0 || Rxbuflen > 4096))
X                  Rxbuflen = 4096;
X             /*
X              * If input is not a regular file, force ACK's each 1024
X              *  (A smarter strategey could be used here ...)
X              */
X             fstat(fileno(in), &f);
X             if (((f.st_mode & S_IFMT) != S_IFREG)
X               && (Rxbuflen == 0 || Rxbuflen > 1024))
X                  Rxbuflen = 1024;
X             vfile("Rxbuflen=%d", Rxbuflen);
X
X             return (sendzsinit());
X         case ZCAN:
X         case TIMEOUT:
X             return ERROR;
X         case ZRQINIT:
X             if (Rxhdr[ZF0] == ZCOMMAND)
X                  continue;
X         default:
X             zshhdr(ZNAK, Txhdr);
X             continue;
```

```
X        }
X    }
X    return ERROR;
X}
X
X/* Send send-init information */
Xsendzsinit()
X{
X    register c;
X    register errors;
X
X    if (Myattn[0] == '\0')
X        return OK;
X    errors = 0;
X    for (;;) {
X        stohdr(0L);
X        zsbhdr(ZSINIT, Txhdr);
X        zsdata(Myattn, 1+strlen(Myattn), ZCRCW);
X        c = zgethdr(Rxhdr, 1);
X        switch (c) {
X        case ZCAN:
X            return ERROR;
X        case ZACK:
X            return OK;
X        default:
X            if (++errors > 9)
X                return ERROR;
X            continue;
X        }
X    }
X}
X
X/* Send file name and related info */
Xzsendfile(buf, blen)
Xchar *buf;
X{
X    register c;
X
X    for (;;) {
X        Txhdr[ZF0] = Lzconv;   /* file conversion request */
X        Txhdr[ZF1] = Lzmanag;  /* file management request */
X        Txhdr[ZF2] = Lztrans;  /* file transport request */
X        Txhdr[ZF3] = 0;
X        zsbhdr(ZFILE, Txhdr);
X        zsdata(buf, blen, ZCRCW);
Xagain:
X        c = zgethdr(Rxhdr, 1);
X        switch (c) {
X        case ZRINIT:
X            goto again;
X        case ZCAN:
X        case TIMEOUT:
X        case ZABORT:
```

```
X         case ZFIN:
X              return ERROR;
X         case ZSKIP:
X              fclose(in); return c;
X         case ZRPOS:
X              fseek(in, Rxpos, 0);
X              Txpos = Rxpos; Lastc = -1; Dontread = FALSE;
X              return zsendfdata();
X         default:
X              continue;
X         }
X    }
X}
X
X/* Send the data in the file */
Xzsendfdata()
X{
X    register c, e;
X    register newcnt;
X    register long tcount = 0;
X    static int tleft = 6;   /* Counter for test mode */
X    extern int setjmp();
X
X    if (Baudrate > 300)
X         blklen = 256;
X    if (Baudrate > 2400)
X         blklen = KSIZE;
X/*  if (Rxbuflen && blklen>Rxbuflen) */
X    if (Rxbuflen)
X         blklen = Rxbuflen;
X    if (blkopt && blklen > blkopt)
X         blklen = blkopt;
X    vfile("Rxbuflen=%d blklen=%d", Rxbuflen, blklen);
Xsomemore:
X    if (setjmp(intrjmp)) {
Xwaitack:
X         c = getinsync();
X         switch (c) {
X         default:
X         case ZCAN:
X              fclose(in);
X              return ERROR;
X         case ZSKIP:
X              fclose(in);
X              return c;
X         case ZACK:
X         case ZRPOS:
X              break;
X         case ZRINIT:
X              return OK;
X         }
X#ifdef READCHECK
X         /*
```

```
X          * If the reverse channel can be tested for data,
X          *  this logic may be used to detect error packets
X          *  sent by the receiver, in place of setjmp/longjmp
X          *  rdchk(fdes) returns non 0 if a character is available
X          */
X         while (rdchk(iofd)) {
X#ifdef SVR2
X              switch (checked)
X#else
X              switch (readline(1))
X#endif
X              {
X              case CAN:
X              case ZPAD:
X                   goto waitack;
X              case XOFF:         /* Wait a while for an XON */
X              case XOFF|0200:
X                   readline(100);
X              }
X         }
X#endif
X    }
X
X    if ( !Fromcu)
X         signal(SIGINT, onintr);
X    newcnt = Rxbuflen;
X    stohdr(Txpos);
X    zsbhdr(ZDATA, Txhdr);
X
X    /*
X     * Special testing mode.  This should force receiver to Attn,ZRPOS
X     *  many times.  Each time the signal should be caught, causing the
X     *  file to be started over from the beginning.
X     */
X    if (Testattn) {
X         if ( --tleft)
X              while (tcount < 20000) {
X                   printf(qbf); fflush(stdout);
X                   tcount += strlen(qbf);
X#ifdef READCHECK
X                   while (rdchk(iofd)) {
X#ifdef SVR2
X                        switch (checked)
X#else
X                        switch (readline(1))
X#endif
X                        {
X                        case CAN:
X                        case ZPAD:
X#ifdef TCFLSH
X                             ioctl(iofd, TCFLSH, 1);
X#endif
X                             goto waitack;
```

```
X                       case XOFF:    /* Wait for XON */
X                       case XOFF|0200:
X                            readline(100);
X                       }
X                  }
X#endif
X             }
X         signal(SIGINT, SIG_IGN); canit();
X         sleep(3); purgeline(); mode(0);
X         printf("\nsz: Tcount = %ld\n", tcount);
X         if (tleft) {
X              printf("ERROR: Interrupts Not Caught\n");
X              exit(1);
X         }
X         exit(0);
X    }
X
X    do {
X         if (Dontread) {
X              c = Lastc;
X         } else {
X              c = zfilbuf(txbuf, blklen);
X              Lastread = Txpos;  Lastc = c;
X         }
X         if (Verbose > 10)
X              vfile("Dontread=%d c=%d", Dontread, c);
X         Dontread = FALSE;
X         if (c < blklen)
X              e = ZCRCE;
X         else if (Rxbuflen && (newcnt -= c) <= 0)
X              e = ZCRCW;
X         else
X              e = ZCRCG;
X         zsdata(txbuf, c, e);
X         Txpos += c;
X         if (e == ZCRCW)
X              goto waitack;
X#ifdef READCHECK
X         /*
X          * If the reverse channel can be tested for data,
X          *  this logic may be used to detect error packets
X          *  sent by the receiver, in place of setjmp/longjmp
X          *  rdchk(fdes) returns non 0 if a character is available
X          */
X         fflush(stdout);
X         while (rdchk(iofd)) {
X#ifdef SVR2
X              switch (checked)
X#else
X              switch (readline(1))
X#endif
X              {
X              case CAN:
```

```
X             case ZPAD:
X#ifdef TCFLSH
X                  ioctl(iofd, TCFLSH, 1);
X#endif
X                  /* zcrce - dinna wanna start a ping-pong game */
X                  zsdata(txbuf, 0, ZCRCE);
X                  goto waitack;
X             case XOFF:         /* Wait a while for an XON */
X             case XOFF|0200:
X                  readline(100);
X             }
X        }
X#endif
X    } while (c == blklen);
X    if ( !Fromcu)
X         signal(SIGINT, SIG_IGN);
X
X    for (;;) {
X         stohdr(Txpos);
X         zsbhdr(ZEOF, Txhdr);
X         switch (getinsync()) {
X         case ZACK:
X              continue;
X         case ZRPOS:
X              goto somemore;
X         case ZRINIT:
X              return OK;
X         case ZSKIP:
X              fclose(in);
X              return c;
X         default:
X              fclose(in);
X              return ERROR;
X         }
X    }
X}
X
X/*
X * Respond to receiver's complaint, get back in sync with receiver
X */
Xgetinsync()
X{
X    register c;
X
X    for (;;) {
X         if (Testattn) {
X              printf("\r\n\n\n***** Signal Caught *****\r\n");
X              Rxpos = 0; c = ZRPOS;
X         } else
X              c = zgethdr(Rxhdr, 0);
X         switch (c) {
X         case ZCAN:
X         case ZABORT:
```

```
X         case ZFIN:
X         case TIMEOUT:
X              return ERROR;
X         case ZRPOS:
X              if (Lastc >= 0 && Lastread == Rxpos) {
X                   Dontread = TRUE;
X              } else {
X                   clearerr(in); /* In case file EOF seen */
X                   fseek(in, Rxpos, 0);
X              }
X              Txpos = Rxpos;
X              return c;
X         case ZACK:
X              return c;
X         case ZRINIT:
X         case ZSKIP:
X              fclose(in);
X              return c;
X         case ERROR:
X         default:
X              zsbhdr(ZNAK, Txhdr);
X              continue;
X         }
X    }
X}
X/* Say "bibi" to the receiver, try to do it cleanly */
Xsaybibi()
X{
X    for (;;) {
X         stohdr(0L);
X         zsbhdr(ZFIN, Txhdr);
X         switch (zgethdr(Rxhdr, 0)) {
X         case ZFIN:
X              sendline('O'); sendline('O'); flushmo();
X         case ZCAN:
X         case TIMEOUT:
X              return;
X         }
X    }
X}
X
X/* Local screen character display function */
Xbttyout(c)
X{
X    if (Verbose)
X         putc(c, stderr);
X}
X
X/* Send command and related info */
Xzsendcmd(buf, blen)
Xchar *buf;
X{
X    register c, errors;
```

```
X    long cmdnum;
X
X    cmdnum = getpid();
X    errors = 0;
X    for (;;) {
X         stohdr(cmdnum);
X         Txhdr[ZF0] = Cmdack1;
X         zsbhdr(ZCOMMAND, Txhdr);
X         zsdata(buf, blen, ZCRCW);
Xlisten:
X         Rxtimeout = 100;       /* Ten second wait for resp. */
X         c = zgethdr(Rxhdr, 1);
X
X         switch (c) {
X         case ZRINIT:
X              continue;
X         case ERROR:
X         case TIMEOUT:
X              if (++errors > Cmdtries)
X                   return ERROR;
X              continue;
X         case ZCAN:
X         case ZABORT:
X         case ZFIN:
X         case ZSKIP:
X         case ZRPOS:
X              return ERROR;
X         default:
X              if (++errors > 10)
X                   return ERROR;
X              continue;
X         case ZCOMPL:
X              Exitcode = Rxpos;
X              saybibi();
X              return OK;
X         case ZRQINIT:
X              vfile("******** RZ *******");
X              system("rz");
X              vfile("******** SZ *******");
X              goto listen;
X         }
X    }
X}
X
X/*
X * If called as sb use YMODEM protocol
X */
Xchkinvok(s)
Xchar *s;
X{
X    register char *p;
X
X    p = s;
```

```
X    while (*p == '-')
X         s = ++p;
X    while (*p)
X         if (*p++ == '/')
X              s = p;
X    if (*s == 'v') {
X         Verbose=1; ++s;
X    }
X    Progname = s;
X    if (s[0]=='s' && s[1]=='b') {
X         Nozmodem = TRUE; blklen=KSIZE;
X    }
X}
X
SHAR_EOF
chmod 0600 sz.c || echo "restore of sz.c fails"
sed 's/^X//' << 'SHAR_EOF' > zm.c &&
X/*
X *   Z M . C
X *    ZMODEM protocol primitives
X *    01-19-87  Chuck Forsberg Omen Technology Inc
X *
X * Entry point Functions:
X *  zsbhdr(type, hdr) send binary header
X *  zshhdr(type, hdr) send hex header
X *  zgethdr(hdr, eflag) receive header - binary or hex
X *  zsdata(buf, len, frameend) send data
X *  zrdata(buf, len) receive data
X *  stohdr(pos) store position data in Txhdr
X *  long rclhdr(hdr) recover position offset from header
X */
X
X#ifndef CANFDX
X#include "zmodem.h"
Xint Rxtimeout = 100;       /* Tenths of seconds to wait for something */
X#endif
X
Xstatic char *frametypes[] = {
X    "Carrier Lost",        /* -3 */
X    "TIMEOUT",         /* -2 */
X    "ERROR",      /* -1 */
X#define FTOFFSET 3
X    "ZRQINIT",
X    "ZRINIT",
X    "ZSINIT",
X    "ZACK",
X    "ZFILE",
X    "ZSKIP",
X    "ZNAK",
X    "ZABORT",
X    "ZFIN",
X    "ZRPOS",
X    "ZDATA",
```

```
X    "ZEOF",
X    "ZFERR",
X    "ZCRC",
X    "ZCHALLENGE",
X    "ZCOMPL",
X    "ZCAN",
X    "ZFREECNT",
X    "ZCOMMAND",
X    "ZSTDERR",
X    "xxxxx"
X#define FRTYPES 22    /* Total number of frame types in this array */
X             /*  not including psuedo negative entries */
X};
X
X/* Send ZMODEM binary header hdr of type type */
Xzsbhdr(type, hdr)
Xregister char *hdr;
X{
X    register n;
X    register unsigned short crc;
X
X    vfile("zsbhdr: %s %lx", frametypes[type+FTOFFSET], rclhdr(hdr));
X    if (type == ZDATA)
X        for (n = Znulls; --n >=0; )
X            zsendline(0);
X
X    xsendline(ZPAD); xsendline(ZDLE);
X
X    if (Txfcs32)
X        zsbh32(hdr, type);
X    else {
X        xsendline(ZBIN); zsendline(type); crc = updcrc(type, 0);
X
X        for (n=4; --n >= 0;) {
X            zsendline(*hdr);
X            crc = updcrc((0377& *hdr++), crc);
X        }
X        crc = updcrc(0,updcrc(0,crc));
X        zsendline(crc>>8);
X        zsendline(crc);
X    }
X    if (type != ZDATA)
X        flushmo();
X}
X
X
X/* Send ZMODEM binary header hdr of type type */
Xzsbh32(hdr, type)
Xregister char *hdr;
X{
X    register n;
X    register unsigned long crc;
X
```

```
X    xsendline(ZBIN32);  zsendline(type);
X    crc = 0xFFFFFFFF; crc = UPDC32(type, crc);
X
X    for (n=4; --n >= 0;) {
X         zsendline(*hdr);
X         crc = UPDC32((0377& *hdr++), crc);
X    }
X    crc = ~crc;
X    for (n=4; --n >= 0;) {
X         zsendline(crc);  crc >>= 8;
X    }
X}
X
X/* Send ZMODEM HEX header hdr of type type */
Xzshhdr(type, hdr)
Xregister char *hdr;
X{
X    register n;
X    register unsigned short crc;
X
X    vfile("zshhdr: %s %lx", frametypes[type+FTOFFSET], rclhdr(hdr));
X    sendline(ZPAD); sendline(ZPAD); sendline(ZDLE); sendline(ZHEX);
X    zputhex(type);
X
X    crc = updcrc(type, 0);
X    for (n=4; --n >= 0;) {
X         zputhex(*hdr); crc = updcrc((0377& *hdr++), crc);
X    }
X    crc = updcrc(0,updcrc(0,crc));
X    zputhex(crc>>8); zputhex(crc);
X
X    /* Make it printable on remote machine */
X    sendline(015); sendline(012);
X    /*
X     * Uncork the remote in case a fake XOFF has stopped data flow
X     */
X    if (type != ZFIN)
X         sendline(021);
X    flushmo();
X}
X
X/*
X * Send binary array buf of length length, with ending ZDLE sequence frameend
X */
Xzsdata(buf, length, frameend)
Xregister char *buf;
X{
X    register unsigned short crc;
X
X    vfile("zsdata: length=%d end=%x", length, frameend);
X    if (Txfcs32)
X         zsda32(buf, length, frameend);
X    else {
```

```
SHAR_EOF
echo "End of part 4, continue with part 5"
echo "5" > s2_seq_.tmp
exit 0


From 4859657467853744219
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,a23814e18fb9def5,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-27 11:43:23 PST
Path: utzoo!attcan!uunet!lll-winken!lll-tis!ames!umd5!purdue!decwrl!ucbvax!agate!
saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i040: rzsz -- UNIX side for zmdm file transfer program  part05/05
Keywords: UNIX
Message-ID: <3487@saturn.ucsc.edu>
Date: 27 May 88 18:43:23 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 658
Approved: koreth@ssyx.ucsc.edu

Submitted-by: bammi@mandrill.ces.cwru.edu (Jwahar R. Bammi)
Posting-number: Volume 1, Issue 40
Archive-name: rzsz/part05

#!/bin/sh
# this is part 5 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file zm.c continued
#
CurArch=5
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> zm.c
X        crc = 0;
X        for (;--length >= 0;) {
X             zsendline(*buf); crc = updcrc((0377& *buf++), crc);
X        }
X        xsendline(ZDLE); xsendline(frameend);
X        crc = updcrc(frameend, crc);
X
X        crc = updcrc(0,updcrc(0,crc));
X        zsendline(crc>>8); zsendline(crc);
X    }
```

```
X    if (frameend == ZCRCW) {
X         xsendline(XON);  flushmo();
X    }
X}
X
Xzsda32(buf, length, frameend)
Xregister char *buf;
X{
X    register unsigned long crc;
X
X    crc = 0xFFFFFFFF;
X    for (;--length >= 0;) {
X         zsendline(*buf); crc = UPDC32((0377& *buf++), crc);
X    }
X    xsendline(ZDLE); xsendline(frameend);
X    crc = UPDC32(frameend, crc);
X
X    crc = ~crc;
X    for (length=4; --length >= 0;) {
X         zsendline(crc);  crc >>= 8;
X    }
X}
X
X/*
X * Receive array buf of max length with ending ZDLE sequence
X *  and CRC.  Returns the ending character or error code.
X */
Xzrdata(buf, length)
Xregister char *buf;
X{
X    register c;
X    register unsigned short crc;
X    register d;
X
X    if (Rxframeind == ZBIN32)
X         return zrdat32(buf, length);
X
X    crc = Rxcount = 0;
X    for (;;) {
X         if ((c = zdlread()) & ~0377) {
Xcrcfoo:
X              switch (c) {
X              case GOTCRCE:
X              case GOTCRCG:
X              case GOTCRCQ:
X              case GOTCRCW:
X                   crc = updcrc((d=c)&0377, crc);
X                   if ((c = zdlread()) & ~0377)
X                        goto crcfoo;
X                   crc = updcrc(c, crc);
X                   if ((c = zdlread()) & ~0377)
X                        goto crcfoo;
X                   crc = updcrc(c, crc);
```

```
X                  if (crc & 0xFFFF) {
X                       zperr("Bad data CRC %x", crc);
X                       return ERROR;
X                  }
X                  vfile("zrdata: cnt = %d ret = %x", Rxcount, d);
X                  return d;
X             case GOTCAN:
X                  zperr("ZMODEM: Sender Canceled");
X                  return ZCAN;
X             case TIMEOUT:
X                  zperr("ZMODEM data TIMEOUT");
X                  return c;
X             default:
X                  zperr("ZMODEM bad data subpacket ret=%x", c);
X                  return c;
X             }
X        }
X        if (--length < 0) {
X             zperr("ZMODEM data subpacket too long");
X             return ERROR;
X        }
X        ++Rxcount;
X        *buf++ = c;
X        crc = updcrc(c, crc);
X        continue;
X    }
X}
Xzrdat32(buf, length)
Xregister char *buf;
X{
X    register c;
X    register unsigned long crc;
X    register d;
X
X    crc = 0xFFFFFFFF;  Rxcount = 0;
X    for (;;) {
X         if ((c = zdlread()) & ~0377) {
Xcrcfoo:
X             switch (c) {
X             case GOTCRCE:
X             case GOTCRCG:
X             case GOTCRCQ:
X             case GOTCRCW:
X                  crc = UPDC32((d=c)&0377, crc);
X                  if ((c = zdlread()) & ~0377)
X                       goto crcfoo;
X                  crc = UPDC32(c, crc);
X                  if ((c = zdlread()) & ~0377)
X                       goto crcfoo;
X                  crc = UPDC32(c, crc);
X                  if ((c = zdlread()) & ~0377)
X                       goto crcfoo;
X                  crc = UPDC32(c, crc);
```

```
X                 if ((c = zdlread()) & ~0377)
X                      goto crcfoo;
X                 crc = UPDC32(c, crc);
X                 if (crc != 0xDEBB20E3) {
X                      zperr("Bad data CRC %lX", crc);
X                      return ERROR;
X                 }
X                 vfile("zrdat32: cnt = %d ret = %x", Rxcount, d);
X                 return d;
X            case GOTCAN:
X                 zperr("ZMODEM: Sender Canceled");
X                 return ZCAN;
X            case TIMEOUT:
X                 zperr("ZMODEM data TIMEOUT");
X                 return c;
X            default:
X                 zperr("ZMODEM bad data subpacket ret=%x", c);
X                 return c;
X            }
X        }
X        if (--length < 0) {
X            zperr("ZMODEM data subpacket too long");
X            return ERROR;
X        }
X        ++Rxcount;
X        *buf++ = c;
X        crc = UPDC32(c, crc);
X        continue;
X    }
X}
X
X
X/*
X * Read a ZMODEM header to hdr, either binary or hex.
X *  eflag controls local display of non zmodem characters:
X *  0:  no display
X *  1:  display printing characters only
X *  2:  display all non ZMODEM characters
X *  On success, set Zmodem to 1 and return type of header.
X *   Otherwise return negative on error
X */
Xzgethdr(hdr, eflag)
Xchar *hdr;
X{
X    register c, n, cancount;
X
X    n = Baudrate; /* Max characters before start of frame */
X    cancount = 5;
Xagain:
X    Rxframeind = Rxtype = 0;
X    switch (c = noxrd7()) {
X    case RCDO:
X    case TIMEOUT:
```

```
X         goto fifi;
X    case CAN:
X         if (--cancount <= 0) {
X              c = ZCAN; goto fifi;
X         }
X    /* **** FALL THRU TO **** */
X    default:
Xagn2:
X         if ( --n == 0) {
X              zperr("ZMODEM Garbage count exceeded");
X              return(ERROR);
X         }
X         if (eflag && ((c &= 0177) & 0140))
X              bttyout(c);
X         else if (eflag > 1)
X              bttyout(c);
X         if (c != CAN)
X              cancount = 5;
X         goto again;
X    case ZPAD:         /* This is what we want. */
X         break;
X    }
X    cancount = 5;
Xsplat:
X    switch (c = noxrd7()) {
X    case ZPAD:
X         goto splat;
X    case RCDO:
X    case TIMEOUT:
X         goto fifi;
X    default:
X         goto agn2;
X    case ZDLE:         /* This is what we want. */
X         break;
X    }
X
X    switch (c = noxrd7()) {
X    case RCDO:
X    case TIMEOUT:
X         goto fifi;
X    case ZBIN:
X         Rxframeind = ZBIN;
X         c =  zrbhdr(hdr);
X         break;
X    case ZBIN32:
X         Rxframeind = ZBIN32;
X         c =  zrbhdr32(hdr);
X         break;
X    case ZHEX:
X         Rxframeind = ZHEX;
X         c =  zrhhdr(hdr);
X         break;
X    case CAN:
```

```
X         if (--cancount <= 0) {
X              c = ZCAN; goto fifi;
X         }
X         goto agn2;
X    default:
X         goto agn2;
X    }
X    Rxpos = hdr[ZP3] & 0377;
X    Rxpos = (Rxpos<<8) + (hdr[ZP2] & 0377);
X    Rxpos = (Rxpos<<8) + (hdr[ZP1] & 0377);
X    Rxpos = (Rxpos<<8) + (hdr[ZP0] & 0377);
Xfifi:
X    switch (c) {
X    case GOTCAN:
X         c = ZCAN;
X    /* **** FALL THRU TO **** */
X    case ZNAK:
X    case ZCAN:
X    case ERROR:
X    case TIMEOUT:
X    case RCDO:
X         zperr("ZMODEM: Got %s %s", frametypes[c+FTOFFSET],
X           (c >= 0) ? "header" : "error");
X    /* **** FALL THRU TO **** */
X    default:
X         if (c >= -3 && c <= FRTYPES)
X              vfile("zgethdr: %s %lx", frametypes[c+FTOFFSET], Rxpos);
X         else
X              vfile("zgethdr: %d %lx", c, Rxpos);
X    }
X    return c;
X}
X
X/* Receive a binary style header (type and position) */
Xzrbhdr(hdr)
Xregister char *hdr;
X{
X    register c, n;
X    register unsigned short crc;
X
X    if ((c = zdlread()) & ~0377)
X         return c;
X    Rxtype = c;
X    crc = updcrc(c, 0);
X
X    for (n=4; --n >= 0;) {
X         if ((c = zdlread()) & ~0377)
X              return c;
X         crc = updcrc(c, crc);
X         *hdr++ = c;
X    }
X    if ((c = zdlread()) & ~0377)
X         return c;
```

```
X    crc = updcrc(c, crc);
X    if ((c = zdlread()) & ~0377)
X         return c;
X    crc = updcrc(c, crc);
X    if (crc & 0xFFFF) {
X         zperr("Bad Header CRC"); return ERROR;
X    }
X    Zmodem = 1;
X    return Rxtype;
X}
X
X/* Receive a binary style header (type and position) with 32 bit FCS */
Xzrbhdr32(hdr)
Xregister char *hdr;
X{
X    register c, n;
X    register unsigned long crc;
X
X    if ((c = zdlread()) & ~0377)
X         return c;
X    Rxtype = c;
X    crc = 0xFFFFFFFF; crc = UPDC32(c, crc);
X
X    for (n=4; --n >= 0;) {
X         if ((c = zdlread()) & ~0377)
X              return c;
X         crc = UPDC32(c, crc);
X         *hdr++ = c;
X    }
X    for (n=4; --n >= 0;) {
X         if ((c = zdlread()) & ~0377)
X              return c;
X         crc = UPDC32(c, crc);
X    }
X    if (crc != 0xDEBB20E3) {
X         zperr("Bad Header CRC %lX", crc); return ERROR;
X    }
X    Zmodem = 1;
X    return Rxtype;
X}
X
X
X/* Receive a hex style header (type and position) */
Xzrhhdr(hdr)
Xchar *hdr;
X{
X    register c;
X    register unsigned short crc;
X    register n;
X
X    if ((c = zgethex()) < 0)
X         return c;
X    Rxtype = c;
```

```
X    crc = updcrc(c, 0);
X
X    for (n=4; --n >= 0;) {
X         if ((c = zgethex()) < 0)
X              return c;
X         crc = updcrc(c, crc);
X         *hdr++ = c;
X    }
X    if ((c = zgethex()) < 0)
X         return c;
X    crc = updcrc(c, crc);
X    if ((c = zgethex()) < 0)
X         return c;
X    crc = updcrc(c, crc);
X    if (crc & 0xFFFF) {
X         zperr("Bad Header CRC"); return ERROR;
X    }
X    if (readline(1) == '\r')    /* Throw away possible cr/lf */
X         readline(1);
X    Zmodem = 1; return Rxtype;
X}
X
X/* Send a byte as two hex digits */
Xzputhex(c)
Xregister c;
X{
X    static char   digits[] = "0123456789abcdef";
X
X    if (Verbose>4)
X         vfile("zputhex: %x", c);
X    sendline(digits[(c&0xF0)>>4]);
X    sendline(digits[(c)&0xF]);
X}
X
X/*
X * Send character c with ZMODEM escape sequence encoding.
X *  Escape XON, XOFF. Escape CR following @ (Telenet net escape)
X */
Xzsendline(c)
Xregister c;
X{
X    static lastsent;
X
X    switch (c & 0377) {
X    case ZDLE:
X         xsendline(ZDLE);
X         xsendline (lastsent = (c ^= 0100));
X         break;
X    case 015:
X    case 0215:
X         if ((lastsent & 0177) != '@')
X              goto sendit;
X    /* **** FALL THRU TO **** */
```

```
X    case 020:
X    case 021:
X    case 023:
X    case 0220:
X    case 0221:
X    case 0223:
X#ifdef ZKER
X         if (Zctlesc<0)
X              goto sendit;
X#endif
X         xsendline(ZDLE);
X         c ^= 0100;
Xsendit:
X         xsendline(lastsent = c);
X         break;
X    default:
X#ifdef ZKER
X         if (Zctlesc>0 && ! (c & 0140)) {
X              xsendline(ZDLE);
X              c ^= 0100;
X         }
X#endif
X         xsendline(lastsent = c);
X    }
X}
X
X/* Decode two lower case hex digits into an 8 bit byte value */
Xzgethex()
X{
X    register c;
X
X    c = zgeth1();
X    if (Verbose>4)
X         vfile("zgethex: %x", c);
X    return c;
X}
Xzgeth1()
X{
X    register c, n;
X
X    if ((c = noxrd7()) < 0)
X         return c;
X    n = c - '0';
X    if (n > 9)
X         n -= ('a' - ':');
X    if (n & ~0xF)
X         return ERROR;
X    if ((c = noxrd7()) < 0)
X         return c;
X    c -= '0';
X    if (c > 9)
X         c -= ('a' - ':');
X    if (c & ~0xF)
```

```
X        return ERROR;
X    c += (n<<4);
X    return c;
X}
X
X/*
X * Read a byte, checking for ZMODEM escape encoding
X *  including CAN*5 which represents a quick abort
X */
Xzdlread()
X{
X    register c;
X
X    if ((c = readline(Rxtimeout)) != ZDLE)
X        return c;
X    if ((c = readline(Rxtimeout)) < 0)
X        return c;
X    if (c == CAN && (c = readline(Rxtimeout)) < 0)
X        return c;
X    if (c == CAN && (c = readline(Rxtimeout)) < 0)
X        return c;
X    if (c == CAN && (c = readline(Rxtimeout)) < 0)
X        return c;
X    switch (c) {
X    case CAN:
X        return GOTCAN;
X    case ZCRCE:
X    case ZCRCG:
X    case ZCRCQ:
X    case ZCRCW:
X        return (c | GOTOR);
X    case ZRUB0:
X        return 0177;
X    case ZRUB1:
X        return 0377;
X    default:
X        if ((c & 0140) ==  0100)
X            return (c ^ 0100);
X        break;
X    }
X    zperr("Got bad ZMODEM escape sequence %x", c);
X    return ERROR;
X}
X
X/*
X * Read a character from the modem line with timeout.
X *  Eat parity, XON and XOFF characters.
X */
Xnoxrd7()
X{
X    register c;
X
X    for (;;) {
```

```
X        if ((c = readline(Rxtimeout)) < 0)
X            return c;
X        switch (c &= 0177) {
X        case XON:
X        case XOFF:
X            continue;
X        default:
X            return c;
X        }
X    }
X}
X
X/* Store long integer pos in Txhdr */
Xstohdr(pos)
Xlong pos;
X{
X    Txhdr[ZP0] = pos;
X    Txhdr[ZP1] = pos>>8;
X    Txhdr[ZP2] = pos>>16;
X    Txhdr[ZP3] = pos>>24;
X}
X
X/* Recover a long integer from a header */
Xlong
Xrclhdr(hdr)
Xregister char *hdr;
X{
X    register long l;
X
X    l = (hdr[ZP3] & 0377);
X    l = (l << 8) | (hdr[ZP2] & 0377);
X    l = (l << 8) | (hdr[ZP1] & 0377);
X    l = (l << 8) | (hdr[ZP0] & 0377);
X    return l;
X}
X
SHAR_EOF
chmod 0600 zm.c || echo "restore of zm.c fails"
sed 's/^X//' << 'SHAR_EOF' > zmodem.h &&
X/*
X *   Z M O D E M . H     Manifest constants for ZMODEM
X *    application to application file transfer protocol
X *    01-15-87  Chuck Forsberg Omen Technology Inc
X */
X#define ZPAD '*'  /* 052 Padding character begins frames */
X#define ZDLE 030  /* Ctrl-X Zmodem escape - `ala BISYNC DLE */
X#define ZDLEE (ZDLE^0100)  /* Escaped ZDLE as transmitted */
X#define ZBIN 'A'  /* Binary frame indicator */
X#define ZHEX 'B'  /* HEX frame indicator */
X#define ZBIN32 'C'    /* Binary frame with 32 bit FCS */
X
X/* Frame types (see array "frametypes" in zm.c) */
X#define ZRQINIT   0    /* Request receive init */
```

```
X#define ZRINIT    1    /* Receive init */
X#define ZSINIT 2  /* Send init sequence (optional) */
X#define ZACK 3         /* ACK to above */
X#define ZFILE 4        /* File name from sender */
X#define ZSKIP 5        /* To sender: skip this file */
X#define ZNAK 6         /* Last packet was garbled */
X#define ZABORT 7  /* Abort batch transfers */
X#define ZFIN 8         /* Finish session */
X#define ZRPOS 9        /* Resume data trans at this position */
X#define ZDATA 10  /* Data packet(s) follow */
X#define ZEOF 11        /* End of file */
X#define ZFERR 12  /* Fatal Read or Write error Detected */
X#define ZCRC 13        /* Request for file CRC and response */
X#define ZCHALLENGE 14  /* Receiver's Challenge */
X#define ZCOMPL 15 /* Request is complete */
X#define ZCAN 16        /* Other end canned session with CAN*5 */
X#define ZFREECNT 17    /* Request for free bytes on filesystem */
X#define ZCOMMAND 18    /* Command from sending program */
X#define ZSTDERR 19     /* Output to standard error, data follows */
X
X/* ZDLE sequences */
X#define ZCRCE 'h' /* CRC next, frame ends, header packet follows */
X#define ZCRCG 'i' /* CRC next, frame continues nonstop */
X#define ZCRCQ 'j' /* CRC next, frame continues, ZACK expected */
X#define ZCRCW 'k' /* CRC next, ZACK expected, end of frame */
X#define ZRUB0 'l' /* Translate to rubout 0177 */
X#define ZRUB1 'm' /* Translate to rubout 0377 */
X
X/* zdlread return values (internal) */
X/* -1 is general error, -2 is timeout */
X#define GOTOR 0400
X#define GOTCRCE (ZCRCE|GOTOR)   /* ZDLE-ZCRCE received */
X#define GOTCRCG (ZCRCG|GOTOR)   /* ZDLE-ZCRCG received */
X#define GOTCRCQ (ZCRCQ|GOTOR)   /* ZDLE-ZCRCQ received */
X#define GOTCRCW (ZCRCW|GOTOR)   /* ZDLE-ZCRCW received */
X#define GOTCAN    (GOTOR|030)   /* CAN*5 seen */
X
X/* Byte positions within header array */
X#define ZF0  3    /* First flags byte */
X#define ZF1  2
X#define ZF2  1
X#define ZF3  0
X#define ZP0  0    /* Low order 8 bits of position */
X#define ZP1  1
X#define ZP2  2
X#define ZP3  3    /* High order 8 bits of file position */
X
X/* Bit Masks for ZRINIT flags byte ZF0 */
X#define CANFDX    01   /* Rx can send and receive true FDX */
X#define CANOVIO   02   /* Rx can receive data during disk I/O */
X#define CANBRK    04   /* Rx can send a break signal */
X#define CANCRY    010  /* Receiver can decrypt */
X#define CANLZW    020  /* Receiver can uncompress */
```

```
X#define CANFC32   040  /* Receiver can use 32 bit Frame Check */
X
X/* Parameters for ZSINIT frame */
X#define ZATTNLEN 32   /* Max length of attention string */
X
X/* Parameters for ZFILE frame */
X/* Conversion options one of these in ZF0 */
X#define ZCBIN     1    /* Binary transfer - inhibit conversion */
X#define ZCNL 2    /* Convert NL to local end of line convention */
X#define ZCRESUM   3    /* Resume interrupted file transfer */
X/* Management options, one of these in ZF1 */
X#define ZMNEW     1    /* Transfer if source newer or longer */
X#define ZMCRC     2    /* Transfer if different file CRC or length */
X#define ZMAPND    3    /* Append contents to existing file (if any) */
X#define ZMCLOB    4    /* Replace existing file */
X#define ZMSPARS   5    /* Encoding for sparse file */
X#define ZMDIFF    6    /* Transfer if dates or lengths different */
X#define ZMPROT    7    /* Protect destination file */
X/* Transport options, one of these in ZF2 */
X#define ZTLZW     1    /* Lempel-Ziv compression */
X#define ZTCRYPT   2    /* Encryption */
X#define ZTRLE     3    /* Run Length encoding */
X
X/* Parameters for ZCOMMAND frame ZF0 (otherwise 0) */
X#define ZCACK1    1    /* Acknowledge, then do command */
X
Xlong rclhdr();
X
X/* Globals used by ZMODEM functions */
Xint Rxframeind;       /* ZBIN ZBIN32, or ZHEX type of frame received */
Xint Rxtype;       /* Type of header received */
Xint Rxcount;      /* Count of data bytes received */
Xextern Rxtimeout; /* Tenths of seconds to wait for something */
Xchar Rxhdr[4];        /* Received header */
Xchar Txhdr[4];        /* Transmitted header */
Xlong Rxpos;       /* Received file position */
Xlong Txpos;       /* Transmitted file position */
Xint Txfcs32;      /* TURE means send binary frames with 32 bit FCS */
Xint Znulls;       /* Number of nulls to send at beginning of ZDATA hdr */
Xchar Attn[ZATTNLEN+1];/* Attention string rx sends to tx on err */
X
SHAR_EOF
chmod 0600 zmodem.h || echo "restore of zmodem.h fails"
rm -f s2_seq_.tmp
echo "You have unpacked the last part"
exit 0


From -7107882660201479018
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,523617bbcdb871d2,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-29 11:57:34 PST
```

Path: utzoo!utgpu!water!watmath!clyde!att!rutgers!mailrus!ames!umd5!purdue!decwrl!
ucbvax!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i041: zmdm -- terminal program with file transfer part01/08
Keywords: mwc, alcyon, shar
Message-ID: <3503@saturn.ucsc.edu>
Date: 29 May 88 18:57:34 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1555
Approved: koreth@ssyx.ucsc.edu

Submitted-by: bammi@mandrill.ces.cwru.edu (Jwahar R. Bammi)
Posting-number: Volume 1, Issue 41
Archive-name: zmdm/part01

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
# This is part 1 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
#
#    Run the following text with /bin/sh to create:
#      COMMON.C
#      COMMON.H
#      CONFIG.H
#      DECL.H
#      EXPANDAR.C
#      FILEIO.C
#      HI5025.S
#      LNK
#      MAIN.C
#      MAKEFALC.STA
#      MAKEFILE
#      MAKEFILE.ALC
#      MAKEFILE.MAN
#      MAKEFILE.STA
#      MAKEFMAN.STA
#      MANX.ASM
#      PHONE.C
#      RZ.C
#      RZLNK
#      SZ.C
#      SZLNK
#      TRANSFER.C
#      TYME.C
#      UTIL.C
#      YMODEM.DOC
#      ZM.C
#      ZMDM.DOC
#      ZMDM.H
#      ZMODEM.H
#
```

```
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
echo "x - extracting COMMON.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > COMMON.C &&
X/*
X * 		ACKNOWLEDGEMENTS
X *
X *  ZMDM was derived from rz/sz for Unix  posted by
X *  Chuck Forsberg (...!tektronix!reed!omen!caf ). We
X *  thank him for his excellent code, and for giving
X *  us permission to use and distribute his code and
X *  documentation.
X *
X *  Atari St version by:
X *      Jwahar Bammi
X *           usenet: mandrill!bammi@{decvax,sun}.UUCP
X *           csnet:  bammi@mandrill.ces.CWRU.edu
X *           arpa:   bammi@mandrill.ces.CWRU.edu
X *           CompuServe: 71515,155
X */
X
X#include "config.h"
X
X#define IN_COMMON
X
X/*
X * -rev 08-17-86
X * mode function and most of the rest of the system dependent
X * stuff for rb.c and sb.c   This file is #included so the includer
X * can set parameters such as HOWMANY.  See the main file (rz.c/sz.c)
X * for compile instructions.
X */
X
X#ifdef DLIBS
X#include <string.h>
X#endif
X
X#include "zmdm.h"
X#include "zmodem.h"
X
X#ifndef Supexec
X	/* Some versions of osbind don't define Supexec */
X#define Supexec(X) xbios(38,X)
X#endif
X
X#define CRCTABLE
X
X	/* GLOBALS */
Xint Zmodem;	/* ZMODEM protocol requested */
Xint Nozmodem;	/* If invoked as "rb" */
Xint Badclose;	/* Error on last close */
```

```
Xint Batch;
Xint Wcsmask;
Xint Verbose;
Xint Quiet;        /* overrides logic that would otherwise set verbose */
Xint Lleft;        /* number of characters in linbuf */
Xint Readnum; /* Number of bytes to ask for in read() from modem */
Xint Crcflg;
Xint ForceBinary;      /* local binary force override for rz */
X
XFILE *logf;
XFILE *STDERR;
Xint vdebug;  /* set if RDEBUG or SDEBUG */
X
Xchar secbuf[KSIZE];
Xchar linbuf[KSIZE];
X#if (MWC || MANX)          /* File i/o buffer */
Xunsigned char *bufr;  /* In MWC or MANX it is lmalloc()'ed in main.c */
X#else
X#ifndef DYNABUF
Xunsigned char bufr[BBUFSIZ];
X#else
Xunsigned char *bufr;
X#endif /* DYNABUF */
X#endif
Xint fout;
Xint Lastrx;
Xint Firstsec;
Xint Eofseen;      /* indicates cpm eof (^Z) has been received */
Xint errors;
Xlong Bytesleft;       /* number of bytes of incoming file left */
Xlong Modtime;         /* Unix style mod time for incoming file */
Xint Filemode;         /* Unix style mode for incoming file */
Xchar Pathname[PATHLEN];
Xchar *Progname;       /* the name by which we were called */
X
Xint Thisbinary;       /* current file is to be received in bin mode */
Xint Blklen;       /* record length of received packets */
Xchar Lzmanag;         /* Local file management request */
Xchar zconv;       /* ZMODEM file conversion request */
Xchar zmanag;      /* ZMODEM file management request */
Xchar ztrans;      /* ZMODEM file transport request */
X
Xjmp_buf tohere;       /* For the interrupt on RX timeout */
Xjmp_buf abrtjmp;  /* for force abort */
Xjmp_buf busjmp;       /* for bus errors */
Xjmp_buf addrjmp;  /* for address errors */
Xunsigned long BusErr, AddrErr;  /* saved vector addresses */
Xint Zctlesc;      /* Encode control characters */
Xint SendType;         /* Which send line to use   */
Xint Modem;        /* Send using Xmodem */
Xint lsct;
Xint tryzhdrtype;  /* Header type to send corresponding to Last rx close */
Xint Txfcs32;      /* TRUE means send binary frames with 32 bit FCS */
```

```
X
X    /* Globals used by ZMODEM functions */
Xint Rxframeind;       /* ZBIN or ZHEX indicates type of frame received */
Xint Rxtype;       /* Type of header received */
Xint Rxcount;      /* Count of data bytes received */
Xint Rxtimeout;        /* Tenths of seconds to wait for something */
Xint Znulls;       /* Number of nulls to send at beginning of ZDATA hdr */
Xchar Rxhdr[4];        /* Received header */
Xchar Txhdr[4];        /* Transmitted header */
Xlong Rxpos;       /* Received file position */
Xlong Txpos;       /* Transmitted file position */
Xchar Attn[ZATTNLEN+1];/* Attention string rx sends to tx on err */
X
X        /* Globals used by Sz */
Xchar Lzconv; /* Local ZMODEM file conversion request */
Xchar Lztrans;
Xchar *Cmdstr;         /* Pointer to the command string */
Xint Optiong;      /* Let it rip no wait for sector ACK's */
Xint Noeofseen;
Xint Totsecs;      /* total number of sectors this file */
Xint Command;      /* Send a command, then exit. */
Xint Cmdack1;      /* Rx ACKs command, then do it */
Xint Exitcode;
Xint Testattn;         /* Force receiver to send Attn, etc with qbf. */
Xint Lastc;        /* Count of last buffer read or -1 */
Xint Dontread;         /* Don't read the buffer, it's still there */
Xint Wantfcs32;        /* want to send 32 bit FCS */
Xlong Lastread;        /* Beginning offset of last buffer read */
Xjmp_buf intrjmp;  /* For the interrupt on RX CAN */
X
Xchar *qbf;
Xunsigned int Rxbuflen;/* Receiver's max buffer length */
Xint Cmdtries;
Xint Filcnt;       /* count of number of files opened */
Xint Lfseen;
Xint Tframlen;         /* Override for tx frame length */
Xint blkopt;       /* Override value for zmodem blklen */
Xint Rxflags;
Xint Ascii;        /* Add CR's for brain damaged programs */
Xint Fullname;         /* transmit full pathname */
Xint Unlinkafter;  /* Unlink file after it is sent */
Xint Dottoslash;       /* Change foo.bar.baz to foo/bar/baz */
Xint errcnt;       /* number of files unreadable */
Xint siggi;        /* line interrupt enable flag     */
X
X/* crctab calculated by Mark G. Mendel, Network Systems Corporation */
Xunsigned int crctab[256] = {
X    0x0000,  0x1021,  0x2042,  0x3063,  0x4084,  0x50a5,  0x60c6,  0x70e7,
X    0x8108,  0x9129,  0xa14a,  0xb16b,  0xc18c,  0xd1ad,  0xe1ce,  0xf1ef,
X    0x1231,  0x0210,  0x3273,  0x2252,  0x52b5,  0x4294,  0x72f7,  0x62d6,
X    0x9339,  0x8318,  0xb37b,  0xa35a,  0xd3bd,  0xc39c,  0xf3ff,  0xe3de,
X    0x2462,  0x3443,  0x0420,  0x1401,  0x64e6,  0x74c7,  0x44a4,  0x5485,
X    0xa56a,  0xb54b,  0x8528,  0x9509,  0xe5ee,  0xf5cf,  0xc5ac,  0xd58d,
```

```
X    0x3653,  0x2672,  0x1611,  0x0630,  0x76d7,  0x66f6,  0x5695,  0x46b4,
X    0xb75b,  0xa77a,  0x9719,  0x8738,  0xf7df,  0xe7fe,  0xd79d,  0xc7bc,
X    0x48c4,  0x58e5,  0x6886,  0x78a7,  0x0840,  0x1861,  0x2802,  0x3823,
X    0xc9cc,  0xd9ed,  0xe98e,  0xf9af,  0x8948,  0x9969,  0xa90a,  0xb92b,
X    0x5af5,  0x4ad4,  0x7ab7,  0x6a96,  0x1a71,  0x0a50,  0x3a33,  0x2a12,
X    0xdbfd,  0xcbdc,  0xfbbf,  0xeb9e,  0x9b79,  0x8b58,  0xbb3b,  0xab1a,
X    0x6ca6,  0x7c87,  0x4ce4,  0x5cc5,  0x2c22,  0x3c03,  0x0c60,  0x1c41,
X    0xedae,  0xfd8f,  0xcdec,  0xddcd,  0xad2a,  0xbd0b,  0x8d68,  0x9d49,
X    0x7e97,  0x6eb6,  0x5ed5,  0x4ef4,  0x3e13,  0x2e32,  0x1e51,  0x0e70,
X    0xff9f,  0xefbe,  0xdfdd,  0xcffc,  0xbf1b,  0xaf3a,  0x9f59,  0x8f78,
X    0x9188,  0x81a9,  0xb1ca,  0xa1eb,  0xd10c,  0xc12d,  0xf14e,  0xe16f,
X    0x1080,  0x00a1,  0x30c2,  0x20e3,  0x5004,  0x4025,  0x7046,  0x6067,
X    0x83b9,  0x9398,  0xa3fb,  0xb3da,  0xc33d,  0xd31c,  0xe37f,  0xf35e,
X    0x02b1,  0x1290,  0x22f3,  0x32d2,  0x4235,  0x5214,  0x6277,  0x7256,
X    0xb5ea,  0xa5cb,  0x95a8,  0x8589,  0xf56e,  0xe54f,  0xd52c,  0xc50d,
X    0x34e2,  0x24c3,  0x14a0,  0x0481,  0x7466,  0x6447,  0x5424,  0x4405,
X    0xa7db,  0xb7fa,  0x8799,  0x97b8,  0xe75f,  0xf77e,  0xc71d,  0xd73c,
X    0x26d3,  0x36f2,  0x0691,  0x16b0,  0x6657,  0x7676,  0x4615,  0x5634,
X    0xd94c,  0xc96d,  0xf90e,  0xe92f,  0x99c8,  0x89e9,  0xb98a,  0xa9ab,
X    0x5844,  0x4865,  0x7806,  0x6827,  0x18c0,  0x08e1,  0x3882,  0x28a3,
X    0xcb7d,  0xdb5c,  0xeb3f,  0xfb1e,  0x8bf9,  0x9bd8,  0xabbb,  0xbb9a,
X    0x4a75,  0x5a54,  0x6a37,  0x7a16,  0x0af1,  0x1ad0,  0x2ab3,  0x3a92,
X    0xfd2e,  0xed0f,  0xdd6c,  0xcd4d,  0xbdaa,  0xad8b,  0x9de8,  0x8dc9,
X    0x7c26,  0x6c07,  0x5c64,  0x4c45,  0x3ca2,  0x2c83,  0x1ce0,  0x0cc1,
X    0xef1f,  0xff3e,  0xcf5d,  0xdf7c,  0xaf9b,  0xbfba,  0x8fd9,  0x9ff8,
X    0x6e17,  0x7e36,  0x4e55,  0x5e74,  0x2e93,  0x3eb2,  0x0ed1,  0x1ef0
X};
X
X/*
X * Copyright (C) 1986 Gary S. Brown.  You may use this program, or
X * code or tables extracted from it, as desired without restriction.
X *
X *   Need an unsigned type capable of holding 32 bits;
X */
Xtypedef unsigned long int UNS_32_BITS;
X
X/* First, the polynomial itself and its table of feedback terms.  The  */
X/* polynomial is                                                       */
X/* X^32+X^26+X^23+X^22+X^16+X^12+X^11+X^10+X^8+X^7+X^5+X^4+X^2+X^1+X^0 */
X/* Note that we take it "backwards" and put the highest-order term in  */
X/* the lowest-order bit.  The X^32 term is "implied"; the LSB is the   */
X/* X^31 term, etc.  The X^0 term (usually shown as "+1") results in    */
X/* the MSB being 1.                                                    */
X
X/* Note that the usual hardware shift register implementation, which   */
X/* is what we're using (we're merely optimizing it by doing eight-bit  */
X/* chunks at a time) shifts bits into the lowest-order term.  In our   */
X/* implementation, that means shifting towards the right.  Why do we   */
X/* do it this way?  Because the calculated CRC must be transmitted in  */
X/* order from highest-order term to lowest-order term.  UARTs transmit */
X/* characters in order from LSB to MSB.  By storing the CRC this way,  */
X/* we hand it to the UART in the order low-byte to high-byte; the UART */
X/* sends each low-bit to hight-bit; and the result is transmission bit */
```

```
X/* by bit from highest- to lowest-order term without requiring any bit */
X/* shuffling on our part.  Reception works similarly.                   */
X
X/* The feedback terms table consists of 256, 32-bit entries.  Notes:   */
X/*                                                                      */
X/*     The table can be generated at runtime if desired; code to do so */
X/*     is shown later.  It might not be obvious, but the feedback      */
X/*     terms simply represent the results of eight shift/xor opera-    */
X/*     tions for all combinations of data and CRC register values.     */
X/*                                                                      */
X/*     The values must be right-shifted by eight bits by the "updcrc"  */
X/*     logic; the shift must be unsigned (bring in zeroes).  On some   */
X/*     hardware you could probably optimize the shift in assembler by  */
X/*     using byte-swap instructions.                                    */
X
Xunsigned long int crc_32_tab[] = { /* CRC polynomial 0xedb88320 */
X    0x00000000L, 0x77073096L, 0xee0e612cL, 0x990951baL,
X    0x076dc419L, 0x706af48fL, 0xe963a535L, 0x9e6495a3L,
X    0x0edb8832L, 0x79dcb8a4L, 0xe0d5e91eL, 0x97d2d988L,
X    0x09b64c2bL, 0x7eb17cbdL, 0xe7b82d07L, 0x90bf1d91L,
X    0x1db71064L, 0x6ab020f2L, 0xf3b97148L, 0x84be41deL,
X    0x1adad47dL, 0x6ddde4ebL, 0xf4d4b551L, 0x83d385c7L,
X    0x136c9856L, 0x646ba8c0L, 0xfd62f97aL, 0x8a65c9ecL,
X    0x14015c4fL, 0x63066cd9L, 0xfa0f3d63L, 0x8d080df5L,
X    0x3b6e20c8L, 0x4c69105eL, 0xd56041e4L, 0xa2677172L,
X    0x3c03e4d1L, 0x4b04d447L, 0xd20d85fdL, 0xa50ab56bL,
X    0x35b5a8faL, 0x42b2986cL, 0xdbbbc9d6L, 0xacbcf940L,
X    0x32d86ce3L, 0x45df5c75L, 0xdcd60dcfL, 0xabd13d59L,
X    0x26d930acL, 0x51de003aL, 0xc8d75180L, 0xbfd06116L,
X    0x21b4f4b5L, 0x56b3c423L, 0xcfba9599L, 0xb8bda50fL,
X    0x2802b89eL, 0x5f058808L, 0xc60cd9b2L, 0xb10be924L,
X    0x2f6f7c87L, 0x58684c11L, 0xc1611dabL, 0xb6662d3dL,
X    0x76dc4190L, 0x01db7106L, 0x98d220bcL, 0xefd5102aL,
X    0x71b18589L, 0x06b6b51fL, 0x9fbfe4a5L, 0xe8b8d433L,
X    0x7807c9a2L, 0x0f00f934L, 0x9609a88eL, 0xe10e9818L,
X    0x7f6a0dbbL, 0x086d3d2dL, 0x91646c97L, 0xe6635c01L,
X    0x6b6b51f4L, 0x1c6c6162L, 0x856530d8L, 0xf262004eL,
X    0x6c0695edL, 0x1b01a57bL, 0x8208f4c1L, 0xf50fc457L,
X    0x65b0d9c6L, 0x12b7e950L, 0x8bbeb8eaL, 0xfcb9887cL,
X    0x62dd1ddfL, 0x15da2d49L, 0x8cd37cf3L, 0xfbd44c65L,
X    0x4db26158L, 0x3ab551ceL, 0xa3bc0074L, 0xd4bb30e2L,
X    0x4adfa541L, 0x3dd895d7L, 0xa4d1c46dL, 0xd3d6f4fbL,
X    0x4369e96aL, 0x346ed9fcL, 0xad678846L, 0xda60b8d0L,
X    0x44042d73L, 0x33031de5L, 0xaa0a4c5fL, 0xdd0d7cc9L,
X    0x5005713cL, 0x270241aaL, 0xbe0b1010L, 0xc90c2086L,
X    0x5768b525L, 0x206f85b3L, 0xb966d409L, 0xce61e49fL,
X    0x5edef90eL, 0x29d9c998L, 0xb0d09822L, 0xc7d7a8b4L,
X    0x59b33d17L, 0x2eb40d81L, 0xb7bd5c3bL, 0xc0ba6cadL,
X    0xedb88320L, 0x9abfb3b6L, 0x03b6e20cL, 0x74b1d29aL,
X    0xead54739L, 0x9dd277afL, 0x04db2615L, 0x73dc1683L,
X    0xe3630b12L, 0x94643b84L, 0x0d6d6a3eL, 0x7a6a5aa8L,
X    0xe40ecf0bL, 0x9309ff9dL, 0x0a00ae27L, 0x7d079eb1L,
X    0xf00f9344L, 0x8708a3d2L, 0x1e01f268L, 0x6906c2feL,
```

```
X    0xf762575dL, 0x806567cbL, 0x196c3671L, 0x6e6b06e7L,
X    0xfed41b76L, 0x89d32be0L, 0x10da7a5aL, 0x67dd4accL,
X    0xf9b9df6fL, 0x8ebeeff9L, 0x17b7be43L, 0x60b08ed5L,
X    0xd6d6a3e8L, 0xa1d1937eL, 0x38d8c2c4L, 0x4fdff252L,
X    0xd1bb67f1L, 0xa6bc5767L, 0x3fb506ddL, 0x48b2364bL,
X    0xd80d2bdaL, 0xaf0a1b4cL, 0x36034af6L, 0x41047a60L,
X    0xdf60efc3L, 0xa867df55L, 0x316e8eefL, 0x4669be79L,
X    0xcb61b38cL, 0xbc66831aL, 0x256fd2a0L, 0x5268e236L,
X    0xcc0c7795L, 0xbb0b4703L, 0x220216b9L, 0x5505262fL,
X    0xc5ba3bbeL, 0xb2bd0b28L, 0x2bb45a92L, 0x5cb36a04L,
X    0xc2d7ffa7L, 0xb5d0cf31L, 0x2cd99e8bL, 0x5bdeae1dL,
X    0x9b64c2b0L, 0xec63f226L, 0x756aa39cL, 0x026d930aL,
X    0x9c0906a9L, 0xeb0e363fL, 0x72076785L, 0x05005713L,
X    0x95bf4a82L, 0xe2b87a14L, 0x7bb12baeL, 0x0cb61b38L,
X    0x92d28e9bL, 0xe5d5be0dL, 0x7cdcefb7L, 0x0bdbdf21L,
X    0x86d3d2d4L, 0xf1d4e242L, 0x68ddb3f8L, 0x1fda836eL,
X    0x81be16cdL, 0xf6b9265bL, 0x6fb077e1L, 0x18b74777L,
X    0x88085ae6L, 0xff0f6a70L, 0x66063bcaL, 0x11010b5cL,
X    0x8f659effL, 0xf862ae69L, 0x616bffd3L, 0x166ccf45L,
X    0xa00ae278L, 0xd70dd2eeL, 0x4e048354L, 0x3903b3c2L,
X    0xa7672661L, 0xd06016f7L, 0x4969474dL, 0x3e6e77dbL,
X    0xaed16a4aL, 0xd9d65adcL, 0x40df0b66L, 0x37d83bf0L,
X    0xa9bcae53L, 0xdebb9ec5L, 0x47b2cf7fL, 0x30b5ffe9L,
X    0xbdbdf21cL, 0xcabac28aL, 0x53b39330L, 0x24b4a3a6L,
X    0xbad03605L, 0xcdd70693L, 0x54de5729L, 0x23d967bfL,
X    0xb3667a2eL, 0xc4614ab8L, 0x5d681b02L, 0x2a6f2b94L,
X    0xb40bbe37L, 0xc30c8ea1L, 0x5a05df1bL, 0x2d02ef8dL
X};
X
X#ifndef REMOTE
Xint *aline_addr;      /* Base addr of aline variables */
X#endif
X
Xint hlines = 25;      /* # of lines on screen     */
Xint rez;          /* current resolution       */
Xint scolor = 0;            /* current fg/bg screen color toggle */
X
X#ifndef REMOTE
Xlong *ms_ptr;              /* Pointer to my screen memory, aligned at
X                     a 256 bytes boundary */
X#endif /* REMOTE */
X
X#if (!(MWC || MANX))
X#ifndef REMOTE
Xlong m_screen[8*1024+32];  /* My screen memory
X                     32K bytes + 256 Bytes guard for alignement
X                     In the worst case when we align we have to
X                     go 255 bytes from &m_screen[0], hence the
X                     256 Byte guard is required */
X#endif /* REMOTE */
X
X#else /* MWC || MANX */
X
```

```
X#ifdef RECURSE
X#ifdef BIGSTACK
X#ifdef MWC
Xlong _stksize = 128L * 1024L;
X#else
Xlong _STKSIZ = 128L * 1024L;
X#endif
X#else
X#ifdef MWC
Xlong _stksize = 16L * 1024L;
X#else
Xlong _STKSIZ = 16L * 1024L;
X#endif
X#endif /* BIGSTACK */
X#else
X#ifdef MWC
Xlong _stksize = 16384L;
X#else
Xlong _STKSIZ = 16384L;
X#endif
X#endif /* RECURSE */
X
X#ifndef REMOTE
Xlong *m_screen;            /* Presently Mark Willams will not allow
X                     > 32K static structures */
X#endif /* REMOTE */
X
X#endif /* MWC */
X
X#ifdef DLIBS
X#ifdef RECURSE
X#ifdef BIGSTACK
Xlong _STKSIZE = 128L * 1024L;
X#else
Xlong _STKSIZE = 16L * 1024L;
X#endif /* BIGSTACK */
X#else
Xlong _STKSIZE = 16384L;
X#endif /* RECURSE */
X#endif /* DLIBS */
X
Xstruct stat statbuf;         /* Disk Transfer address for Find first etc */
Xint Baudrate;                /* Current baud rate
          */
Xlong drv_map;                /* bit vector of valid drives */
X
XBAUDS vbauds[] = {
X    { "19200", 0, 19200 },
X    { "9600" , 1,  9600 },
X    { "4800" , 2,  4800 },
X    { "4800" , 2,  4800 },
X    { "2400" , 4,  2400 },
X    { "2400" , 4,  2400 },
```

```
X    { "2400" , 4,  2400 },
X    { "1200" , 7,  1200 },
X    { "1200" , 7,  1200 },
X    { "300"  , 9,   300 },
X    { (char *)NULL, -1, -1 }
X};
X
XIOREC save,  /* the original Iorec is saved here for the duration of this
X           process */
X      *savep;     /* ptr returned by Iorec() */
X
Xchar iobuf[IBUFSIZ]; /* My large Rs232 receive buffer */
X
X#ifdef DYNABUF
Xlong BBUFSIZ;
X#endif /* DYNABUF */
X
Xextern long pr_time;
X
X        /* statics */
Xstatic char *tsr_ptr = (char *)0x00fffa2dL; /* See St internals */
Xstatic long *hz_200 =  (long *)0x000004ba; /* Yes the Hitch Hikers */
X                         /* Guide is wrong!! */
X
Xstatic long
X     alrm_time = 0L;     /* Time of next timeout (200 Hz) */
X
Xstatic char *special[] = {
X".PRG", ".TOS", ".TTP", ".ARC", ".ACC", ".IMG", ".RSC", ".O",
X".OBJ", ".NEO",
X".PIC", ".PI1", ".PI2", ".PI3", ".PQ1", ".PQ2", ".PQ3", ".BRD",
X".ANI", ".STW",
X".FNT", ".PRT", ".SNG", ".NEC", ".CNF", ".Z"  , ".DFN", ".GEM",
X".EZD", ".LNK",
X".SYM", ".DVI", ".PIX", ".X32", ".OUT", ".A",   ".CCC", ".CL",
X".CMD", ".COM",
X".CRL", ".DAT", ".DIR", ".EXE", ".OVL", ".PAG", ".REL", ".SAV",
X".SUB", ".SWP",
X".SYS", ".TAR", ".UTL", ".BIN", ".LBR", ".IM",  ".PAK",
X".SBM", ".OBM", ".DIC", ".NDX", ".TFM", ".PBM", ".PBP", ".PCC",
X".PXL",
X""
X};
X
X/*
X * mode(n)
X *  2: set a cbreak, XON/XOFF control mode if using Pro-YAM's -g
X option
X *  1: save old tty stat, set raw mode
X *  0: restore original tty mode
X */
Xmode(n)
X{
```

```
X
X    vfile("mode:%d", n);
X    switch(n) {
X
X    case 2:  /* Cbreak mode used by sb when -g detected */
X         return OK;
X    case 1:
X         return OK;
X    case 0:
X         while(Bconstat(1)) Bconin(1);    /* flush input              */
X
X         return OK;
X    default:
X         return ERROR;
X    }
X}
X
X/*
X * send a break
X * Modifies Bit 3 in the TSR (reg 23) of the Mfp
X */
X
Xsendbrk()
X{
X    register long save_ssp;
X    register long time;
X
X    save_ssp = Super(0L);  /* Super Mode */
X
X    /* set bit 3 of the TSR */
X    *tsr_ptr |= (char)8;
X
X
X    /* wait for 250 ms */
X    time = *hz_200 + 50;
X    while(*hz_200 < time)
X         /* wait */ ;
X
X    /* reset bit 3 of the tsr */
X    *tsr_ptr &= (char)~8;
X
X    Super(save_ssp);  /* Back to user Mode */
X}
X
X#ifdef TX
X/*
X * Put transmit section of the MFP uart off
X *
X */
XTxoff()
X{
X    register long save_ssp;
X
```

```
X    save_ssp = Super(0L);   /* Super Mode */
X
X    /* clear bit 0 of the tsr */
X    *tsr_ptr &= (char)~1;
X
X    Super(save_ssp);   /* Back to user Mode */
X}
X
X/*
X * Put transmit section of the MFP uart on
X *
X */
XTxon()
X{
X    register long save_ssp;
X
X    save_ssp = Super(0L);   /* Super Mode */
X
X    /* set bit 0 of the tsr */
X    *tsr_ptr |= (char)1;
X
X    Super(save_ssp);   /* Back to user Mode */
X}
X#endif /* TX */
X
X
X/*
X * read_modem() - read upto count characters from the modem port
X * Check for user abort (^C) and timeout at the same time
X */
Xint read_modem(buf, count)
Xregister char *buf;
Xregister int count;
X{
X    register int n;
X    extern void rd_time();
X
X    n = 0;
X
X    while(1)
X    {
X#ifndef REMOTE
X        if(Bconstat(2))
X        {
X             /* Character Hit at the Keyboard - is it ^C */
X             if((Bconin(2) & 0x7f) == CTRL('C'))
X             {
X                 /* It is a Control-C */
X                  if(SendType)
X                       bibis(3);
X                  else
X                       bibi(3);
X             }
```

```
X
X         }
X#endif
X         if(Bconstat(1))
X         {
X              /* Character available at Modem Port */
X              n++;
X              *buf++ = Bconin(1);
X
X              while((n < count)  && Bconstat(1))
X              {
X                   *buf++ = Bconin(1);
X                   n++;
X
X              }
X              return n;
X         }
X
X         /* Check for time out if required */
X         if(alrm_time != 0)
X         {
X              /* Alarm Set */
X              Supexec(rd_time);
X              if(pr_time >= alrm_time)
X                  /* timeout */
X                  longjmp(tohere, -1);
X         }
X    }
X}
X
X
X/*
X * alarm() - set the alarm time
X */
Xvoid stalarm(n)
Xunsigned int n;
X{
X    extern void rd_time();
X
X    if(n > 0)
X    {
X         Supexec(rd_time);
X         /* We really need n * 200 but n * 256 is close enough */
X         alrm_time = pr_time + ( n << 8 );
X    }
X    else
X        alrm_time = 0L;
X}
X
X/*
X * write_modem() - send buffer to the modem port
X *
X*/
```

```
Xvoid write_modem(buf,len)
Xregister char *buf;
Xregister int len;
X{
X    register int i;
X    for(i = 0; i < len; i++)
X         Bconout(1, *buf++);
X}
X
X
X/*
X * flushinput() - flush any characters in the modem port
X * a future enhancement may be flush any characters that are in there
X * and any that come in over the next 1 second -- Ymodem does that
X *
X */
Xvoid flushinput()
X{
X    while(Bconstat(1))
X        Bconin(1);
X}
X
X
X
Xint isbinary(name)
Xregister char *name;
X{
X    register char **p;
X    register char *ext;
X    extern char *rindex();
X    extern int ustrcmp();
X
X    if((ext = rindex(name,'.')) == (char *)NULL)
X    {
X         return 0;
X    }
X
X    for(p = special; **p != '\0'; p++)
X        if(ustrcmp(ext,*p) == 0)
X         return 1;
X    return 0;
X}
X
X#define upper(X) (islower(X)?toupper(X):X)
X
X/* Strcmp - case insensative */
Xint ustrcmp(s1,s2)
Xregister char *s1, *s2;
X{
X    while(*s1 != '\0')
X    {
X         if(*s2 == '\0')
X             return 1;
```

```
X        if(upper(*s1) != *s2)
X            return 1;
X        s1++;
X        s2++;
X    }
X    if(*s2 != '\0')
X        return 1;
X    else
X        return 0;
X}
X
X
X/*
X * Ensure that each subdirectory in the pathname exists.
X * If one doesn'nt, make the rest of the directory
X * Returns ERROR if it has trouble creating a path
X *
X */
Xint pathensure(name)
Xchar *name;
X{
X    extern char *index();
X
X    if(index(name, '\\') == (char *)NULL)
X        /* nothing to check */
X        return OK;
X
X    return pathrest(name, (char *)NULL);
X}
X
X/*
X * check rest of the path recursively
X *  If any component of the path name is longer than 8/12 characters then
X *  warn.
X *
X */
Xint pathrest(name, prev)
Xchar *name, *prev;
X{
X    char previous[128];
X    char component[13];
X    register char *p, *q, *r, *s;
X    register int warn;
X    extern char *index();
X    extern char *in();
X
X    if((r = index(name, '\\')) == (char *)NULL)
X        /* nothing more to check */
X        return OK;
X
X    /* pick up the next component of the path name, 8/12 chars max for ST */
X    warn = 0;
X    if((s = in(name, r, '.')) == (char *)NULL)
```

```
X    {
X         /* component does'nt contain a dot */
X         if( ((long)r - (long)name) > 8)
X         {
X              warn++;
X              strncpy(component, name, 8);
X              q = &component[8];
X         }
X         else
X         {
X              register int n;
X
X              n = (int)((long)r - (long)name);
X              strncpy(component, name, n);
X              component[n] = '\0';
X              q = &component[(strlen(component))];
X         }
X    }
X    else
X    {
X         /* component contains a dot */
X         if(((long)s - (long)name) > 8)
X         {
X              warn++;
X              strncpy(component, name, 8);
X              q = &component[8];
X         }
X         else
X         {
X              for(p = name, q = component; p != s; )
X                   *q++ = *p++;
X         }
X         *q++ = '.';
X         s++;
X         if(s != r)
X         {
X              if(((long)r - (long)s) > 3)
X              {
X                   warn++;
X                   strncpy(q, s, 3);
X                   q = &q[3];
X              }
X              else
X              {
X                   for(p = s; p != r; )
X                        *q++ = *p++;
X              }
X         }
X    }
X    *q = '\0';
X
X#ifndef REMOTE
X    if(warn)
```

```
X    {
X         fprintf(stderr,
X         "?WARNING - A component of the path is longer than 8/12 chars\n");
X         fprintf(stderr,"%s --> %s\n", name, component);
X    }
X#endif
X    if(prev == (char *)NULL)
X         strcpy(previous, component);
X    else
X    {
X         strcpy(previous, prev);
X         strcat(previous, "\\");
X         strcat(previous, component);
X    }
X
X    if(existd(previous))
X         /* it exists - go do rest */
X         return pathrest(++r, previous);
X
X    /* does not exist,                    */
X    /* make this component and all its children */
X    return makesubtree(++r, previous);
X}
X
X/*
X * make a subtree
X */
Xint makesubtree(name, prev)
Xchar *name, *prev;
X{
X    char previous[128];
X    char component[13];
X    register char *p, *q, *r, *s;
X    register int warn;
X    extern char *index();
X    extern char *in();
X
X    if(Dcreate(prev) != 0)
X    {
X         log2("Trouble trying to create subtree\n");
X         fprintf(STDERR,"%s\n", prev);
X         return ERROR;
X    }
X#ifndef REMOTE
X    fprintf(stderr,"Created Directory %s\n", prev);
X#endif
X    if((r = index(name, '\\')) == (char *)NULL)
X         /* nothing more to do */
X         return OK;
X
X    /* pick up the next component of the path name, 8/12 chars max for ST */
X    warn = 0;
X    if((s = in(name, r, '.')) == (char *)NULL)
```

```
X    {
X         /* component does'nt contain a dot */
X         if( ((long)r - (long)name) > 8)
X         {
X              warn++;
X              strncpy(component, name, 8);
X              q = &component[8];
X         }
X         else
X         {
X              register int n;
X
X              n = (int)((long)r - (long)name);
X              strncpy(component, name, n);
X              component[n] = '\0';
X              q = &component[(strlen(component))];
X         }
X    }
X    else
X    {
X         /* component contains a dot */
X         if(((long)s - (long)name) > 8)
X         {
X              warn++;
X              strncpy(component, name, 8);
X              q = &component[8];
X         }
X         else
X         {
X              for(p = name, q = component; p != s; )
X                   *q++ = *p++;
X         }
X         *q++ = '.';
X         s++;
X         if(s != r)
X         {
X              if(((long)r - (long)s) > 3)
X              {
X                   warn++;
X                   strncpy(q, s, 3);
X                   q = &q[3];
X              }
X              else
X              {
X                   for(p = s; p != r; )
X                        *q++ = *p++;
X              }
X         }
X    }
X    *q = '\0';
X
X#ifndef REMOTE
X    if(warn)
```

```
X    {
X         fprintf(stderr,
X         "?WARNING - A component of the path is longer than 8/12 chars\n");
X         fprintf(stderr,"%s --> %s\n", name, component);
X    }
X#endif
X    strcpy(previous, prev);
X    strcat(previous, "\\");
X    strcat(previous, component);
X
X
X
X    /* go do the rest of them */
X    return makesubtree(++r, previous);
X}
X
X
X/*
X * test if a subdirectory exists
X * include special case of 'D:\' that Fsfirst does'nt handle correctly
X */
Xint existd(name)
Xregister char *name;
X{
X    /* assuming the DTA buffer is already set up */
X    extern long drv_map;
X    extern struct stat statbuf;
X    register int drive;
X
X    if (Fsfirst(name , 0x0021|0x0010) == 0)
X    {
X         if((statbuf.st_mode & 0x0010) == 0x0010)
X              return TRUE;
X    }
X
X    /* Gemdos doesn't like d:\ style dirs */
X    if((name[3] == '\0') && (name[2] == '\\') && (name[1] == ':'))
X    {
X         drive = name[0];
X         if(isupper(drive))
X              drive = tolower(drive);
X
X         drive = drive - 'a';
X         if((drv_map & (1L << drive)) == 0)
X              return FALSE;
X         else
X              return TRUE;
X    }
X    /* Nor does Gemdos understand '.' or '..' */
X    /* Hey Atari, don't you guys ever test anything */
X    if((strcmp(name,".") == 0) || (strcmp(name,"..") == 0) ||
X       (strcmp(name,".\\") == 0) || (strcmp(name,"..\\") == 0))
X         return TRUE;
```

```
X
X    return FALSE;
X}
X
X/*
X * Does a file exist
X */
Xint existf(name)
Xregister char *name;
X{
X    /* assuming the DTA buffer is already set up */
X
X    return (Fsfirst(name , 0x0001) == 0);
X
X}
X
X/*
X * stsystem(cmd) - execute a process
X * char *cmd;   command to execute (including redirections etc)
X */
Xint stsystem(cmd)
Xchar *cmd;
X{
X        register char *ptr1;             /* general pointers */
X        register char save;              /* */
X        register int status;            /* return status       */
X        register char *args;            /* arguments            */
X        register int len;               /* length of args - 2   */
X        char nils[2];
X        extern char *malloc();
X
X
X        nils[0] = nils[1] = '\0';
X        for(ptr1 = cmd; (!isspace(*ptr1)) && (*ptr1 != '\0'); ptr1++)
X                /* skip till end of path name */;
X        if(*ptr1 != '\0')
X        {
X                /* cmd does have a command tail */
X                /* save the char at the position and terminate the path */
X                save = *ptr1;
X                *ptr1++ = '\0'; /* command tail is the rest of it */
X
X                if((args = (char *) malloc((len = strlen(ptr1)) + 2))
X                    == (char *)NULL)
X                {
X                        /* could not allocate memory */
X                        return(-1);
X                }
X                *args++ = len;
X                strcpy(args,ptr1);
X                args--;
X                /* now do the load and go */
X                status = Pexec(0,cmd,args,(char *)NULL);
```

```
X               /* restore cmd to original state */
X               *--ptr1 = save;
X               free(args);
X       }
X       else
X       {
X               /* command does not have a tail */
X
X               /* now do the load and go */
X               status = Pexec(0,cmd,nils,(char *)NULL);
X       }
X
X       return(status);
X}
X
Xstsleep(n)
Xint n;
X{
X    extern void rd_time();
X
X    if(n != 0)
X    {
X         Supexec(rd_time);
X         /* We really need n * 200 but n * 256 if close enough */
X         alrm_time = pr_time + ( n << 8 );
X
X         while(alrm_time > pr_time)
X         {
X              Supexec(rd_time);
X         }
X         alrm_time = 0L;
X    }
X}
X
X
Xinitz()
X{
X     Zmodem=0;         /* ZMODEM protocol requested */
X     Nozmodem = 0;          /* If invoked as "rb" */
X     Badclose = 0;          /* Error on last close */
X     Batch=0;
X     Wcsmask=0377;
X     Verbose=0;
X     lsct = 1;
X     Quiet=0;     /* overrides logic that would otherwise set verbose */
X     Lleft=0;     /* number of characters in linbuf */
X
X     logf = (FILE *)NULL;
X     vdebug = 0;
X
X     Readnum = KSIZE;  /* Number of bytes to ask for in read() from modem */
X     Crcflg = FALSE;
X     fout = (-1);
```

```
X     errors = 0;
X     Rxtimeout = 100;  /* Tenths of seconds to wait for something */
X     Modem = 0;
X     Blklen = SECSIZ;  /* record length of received packets */
X
X    qbf="The quick brown fox jumped over the lazy dog's back 1234567890\r\n";
X    Cmdtries = 11;
X    Filcnt=0;     /* count of number of files opened */
X    Lfseen=0;
X    Rxbuflen = 1024;   /* Receiver's max buffer length */
X    Tframlen = 0;      /* Override for tx frame length */
X    blkopt=0;     /* Override value for zmodem blklen */
X    Rxflags = 0;
X    Ascii=0;      /* Add CR's for brain damaged programs */
X    Fullname=0;        /* transmit full pathname */
X    Unlinkafter=0;          /* Unlink file after it is sent */
X    Dottoslash=0;      /* Change foo.bar.baz to foo/bar/baz */
X    errcnt=0;     /* number of files unreadable */
X    Testattn = FALSE;
X    siggi = 0;
X    Command = FALSE;
X    Wantfcs32 = TRUE;  /* want to send 32 bit FCS */
X    Znulls = 0;
X    tryzhdrtype=ZRINIT;    /* Header type to send corresponding to Last rx close
*/
X    Txfcs32 = FALSE;   /* TRUE means send binary frames with 32 bit FCS */
X    ForceBinary = FALSE;    /* Local force binary override for rz */
X
X}
X
X/* abort current session - due to async fault */
Xaexit(n)
Xint n;
X{
X    longjmp(abrtjmp, n);
X}
X
Xbuserr()
X{
X    longjmp(busjmp, -1);
X}
X
Xaddrerr()
X{
X    longjmp(addrjmp, -1);
X}
X
X/*
X * Local console output simulation
X */
Xbttyout(c)
X{
X    if (Verbose)
```

```
X#ifdef REMOTE
X        Bconout(1, c);
X#else
X        putc(c, stderr);
X#endif
X}
X
X/*
X *  Send a character to modem.  Small is beautiful.
X */
Xsendline(c)
Xint c;
X{
X#ifndef REMOTE
X    if (Verbose>4)
X    {
X         fprintf(stderr, "Sendline: %x\n", c);
X
X    }
X#endif
X    if (SendType)
X         Bconout(1, c & Wcsmask);
X    else
X         Bconout(1, c);
X
X}
X
X
X
X/*
X * substr(string, token) searches for token in string s
X * returns pointer to token within string if found, NULL otherwise
X */
Xchar *
Xsubstr(s, t)
Xregister char *s;
Xchar *t;
X{
X    register char *ss,*tt;
X    /* search for first char of token */
X    for (ss=s; *s; s++)
X         if (*s == *t)
X              /* compare token with substring */
X              for (ss=s,tt=t; ;)
X              {
X                   if (*tt == '\0')
X                        return s;
X                   if (*ss++ != *tt++)
X                        break;
X              }
X
X    return ((char *)NULL);
X}
```

```
X
X
X/* send cancel string to get the other end to shut up */
Xcanit()
X{
X    static char canistr[] = {
X     ZPAD,ZPAD,24,24,24,24,24,24,24,24,8,8,8,8,8,8,8,8,8,8,0
X    };
X
X    write_modem(canistr, strlen(canistr));
X    if(!SendType)
X         Lleft=0;  /* Do read next time ... */
X
X}
X
X
X/*
X *  Debugging information output interface routine
X */
X/* VARARGS1 */
Xvfile(f, a, b, c)
Xregister char *f;
Xint a, b, c;
X{
X    if (Verbose > 3)
X    {    /* verbose 2 is normal for us */
X         fprintf(STDERR, f, a, b, c);
X         fprintf(STDERR, "\n");
X
X         if(vdebug && (logf != (FILE *)NULL))
X         {
X              fprintf(logf, f, a, b, c);
X              fprintf(logf, "\n");
X              fflush(logf);
X         }
X
X    }
X}
X
X/* longs and pointers are not integers */
X/* VARARGS1 */
Xvfile2(f, a, b, c)
Xlong a, b, c;
Xregister char *f;
X{
X    if (Verbose > 3)
X    {
X         fprintf(STDERR, f, a, b, c);
X         fprintf(STDERR, "\n");
X
X         if(vdebug && (logf != (FILE *)NULL))
X         {
X              fprintf(logf, f, a, b, c);
```

```
X             fprintf(logf, "\n");
X             fflush(logf);
X         }
X
X    }
X}
X
Xchar *in(from, to, c)
Xregister char *from, *to;
Xregister int c;
X{
X    for(; from < to; from++)
X    {
X         if(*from == c)
X              return from;
X    }
X    return (char *)NULL;
X}
X
X/*
X * SetIoBuf() - Save the systems Rs232 buffer and install my large
X * Rs232 buffer.
X *
X */
Xvoid SetIoBuf()
X{
X    /* Get pointer to Rs232 input record */
X    savep = (IOREC *)Iorec(0);
X
X    /* Save the info */
X    save.ibuf= savep->ibuf;
X    save.ibufsiz  = savep->ibufsiz;
X    save.ibufhd   = savep->ibufhd;
X    save.ibuftl   = savep->ibuftl;
X    save.ibuflow  = savep->ibuflow;
X    save.ibufhi   = savep->ibufhi;
X
X    /* Install my buffer in its place */
X    savep->ibuf        = &iobuf[0];
X    savep->ibufsiz     = IBUFSIZ;
X    savep->ibuflow     = IBUFSIZ/4;
X    savep->ibufhi = (IBUFSIZ / 4) * 3;
X    savep->ibufhd = savep->ibuftl = 0;
X
X}
X
X/*
X * ResetIoBuf() - Reset the Rs232 buffer to the saved (system's) buffer
X *
X */
Xvoid ResetIoBuf()
X{
X    savep->ibuf        = save.ibuf;
```

```
X    savep->ibufsiz     = save.ibufsiz;
X    savep->ibuflow     = save.ibuflow;
X    savep->ibufhi = save.ibufhi;
X    savep->ibufhd = save.ibufhd;
X    savep->ibuftl = save.ibuftl;
X}
X
X
Xflush_modem()
X{
X    while(Bcostat(1) == 0);
X}
X
X
X
X/*
X * Log an error if verbose
X */
X
X/*VARARGS1*/
Xlog2(s,p,u)
Xchar *s;
Xint p, u;
X{
X    if (!Verbose)
X         return;
X    fprintf(STDERR, "\nerror %d: ", errors);
X    fprintf(STDERR, s, p, u);
X
X#ifdef RDEBUG
X    if(Verbose > 3)
X    {
X         fprintf(logf, "error %d: ", errors);
X         fprintf(logf, s, p, u);
X    }
X#endif
X
X}
X/*
X * This version of readline is reasonably well suited for
X * reading many characters.
X * timeout is in tenths of seconds
X */
Xint readline(timeout)
Xint timeout;
X{
X    register int n;
X    static char *cdq;  /* pointer for removing chars from linbuf */
X
X    if (--Lleft >= 0)
X    {
X#ifdef RDEBUG
X         if (Verbose > 8)
```

```
X         {
X              fprintf(logf, "%02x ", *cdq&0377);
X         }
X#endif
X
X         return (*cdq++ & Wcsmask);
X    }
X/*  n = timeout/10; */
X    n = timeout >> 3;  /* close enough for rock and roll - see alarm() */
X    if (n < 2)
X         n = 3;
X#ifdef RDEBUG
X    if (Verbose > 3)
X         fprintf(logf, "Calling read: n=%d ", n);
X#endif
X
X    if (setjmp(tohere))
X    {
X         Lleft = 0;
X#ifdef RDEBUG
X         if (Verbose>3)
X         {
X              fprintf(STDERR, "Readline:TIMEOUT\n");
X              fprintf(logf, "Readline:TIMEOUT\n");
X
X         }
X#endif
X         return TIMEOUT;
X    }
X    stalarm(n);
X    Lleft=read_modem(cdq=linbuf, Readnum);
X    stalarm(0);
X
X#ifdef RDEBUG
X    if (Verbose > 3)
X    {
X         fprintf(logf, "Read returned %d bytes\n", Lleft);
X    }
X#endif
X
X    if (Lleft < 1)
X         return TIMEOUT;
X    --Lleft;
X
X#ifdef RDEBUG
X    if (Verbose > 8)
X    {
X         fprintf(logf, "%02x ", *cdq&0377);
X    }
X#endif
X
X    return (*cdq++ & Wcsmask);
X}
```

```
X
Xreport(sct)
Xint sct;
X{
X#ifndef REMOTE
X    if (Verbose>1)
X         fprintf(STDERR,"%03d%c",sct,sct%10? ' ' : '\r');
X#endif
X}
X
Xlreport(sct)
Xlong sct;
X{
X#ifndef REMOTE
X    if (Verbose>1)
X         fprintf(STDERR,"%06ld%c",sct,lsct%10? ' ' : '\r');
X    lsct++;
X#endif
X}
X
X
X#ifdef DLIBS
X_initargs()
X{
X}
X#endif
X
Xwr_modem(s)
Xregister char *s;
X{
X    while(*s != '\0')
X         Bconout(1, *s++);
X}
X
X#ifdef DYNABUF
Xunsigned char *dalloc()
X{
X    register long avail;
X
X    if((avail = (long)Malloc(-1L) - LEAVEALONE) < MINACC)
X         return (unsigned char *)NULL;
X
X    BBUFSIZ = avail;
X    return (unsigned char *)Malloc(BBUFSIZ);
X}
X#endif /* DYNABUF */
X
X/* ********************************************************
X    This code courtesy of katzung@laidbak.UUCP
X     -- thank you very much
X   ******************************************************** */
X
X
```

```
X/* From laidbak!katzung@Sun.COM Tue Apr 26 19:34:41 1988
XReturn-Path: <laidbak!katzung@Sun.COM>
XFrom: Brian Katzung <laidbak!katzung@Sun.COM>
XTo: bammi@mandrill.CES.CWRU.Edu
XSubject: Re:  UW
X*/
X
X#define  MFP  0xFFFFFA01L
X#define  TDDR 36
X#define  hz_200   ((unsigned long *) 0x4BAL)
X
X/* Baud rates 75 (120) and 50 (80) are not currently decoded. */
Xstatic int   timevals[] = {
X    1,   2,   4,   5,   8,   10,  11,
X    16,  32,  64,  96,  128, 143, 175
X};
X
X
X/* load rs232 baud rate, returns -1 if not known */
Xint getbaud ()
X{
X    register int  i;
X    int  tv;           /* Timer value */
X    int  maxtv;             /* Maximum timer value */
X    unsigned long endhz;        /* End of polling period */
X    long savessp;      /* Old stack pointer */
X
X    savessp = Super(0L);
X    maxtv = 0;
X    endhz = *hz_200 + 8;
X    while (*hz_200 < endhz)
X    {
X         tv = *((unsigned char *) (MFP + TDDR));
X         if (tv > maxtv)
X              maxtv = tv;
X    }
X    (void) Super(savessp);
X    for (i = sizeof(timevals) / sizeof(timevals[0]);
X      --i >= 0 && timevals[i] != maxtv; );
X    if (i >= 0)
X         return(i);
X    return(-1);
X}
X
X/* -eof- */
SHAR_EOF
chmod 0600 COMMON.C || echo "restore of COMMON.C fails"
echo "x - extracting COMMON.H (Text)"
sed 's/^X//' << 'SHAR_EOF' > COMMON.H &&
X/*
X *  Common include file
X *
X *      Jwahar Bammi
```

```
X *           usenet: mandrill!bammi@{decvax,sun}.UUCP
X *           csnet:  bammi@mandrill.ces.CWRU.edu
X *           arpa:   bammi@mandrill.ces.CWRU.edu
X *           CompuServe: 71515,155
X */
X
Xextern int Zmodem;         /* ZMODEM protocol requested */
Xextern int Nozmodem;       /* If invoked as "rb" */
Xextern int Badclose;       /* Error on last close */
Xextern int Batch;
Xextern int Wcsmask;
Xextern int Verbose;
Xextern int Quiet;     /* overrides logic that would otherwise set verbose */
Xextern int Lleft;     /* number of characters in linbuf */
Xextern int Readnum;        /* Number of bytes to ask for in read() from modem */
Xextern int Crcflg;
Xextern int ForceBinary;         /* local binary force override for rz */
Xextern char secbuf[];
Xextern char linbuf[];
X#if (MWC || MANX)          /* File i/o buffer */
Xextern unsigned char *bufr;
X#else
X#ifndef DYNABUF
Xextern unsigned char bufr[];
X#else
Xextern unsigned char *bufr;
X#endif
X#endif
Xextern int fout;
Xextern int Lastrx;
Xextern int Firstsec;
Xextern int Eofseen;        /* indicates cpm eof (^Z) has been received */
Xextern int errors;
Xextern long Bytesleft;     /* number of bytes of incoming file left */
Xextern long Modtime;       /* Unix style mod time for incoming file */
Xextern int Filemode;       /* Unix style mode for incoming file */
Xextern char Pathname[];
Xextern char *Progname;     /* the name by which we were called */
Xextern int Zctlesc;        /* Encode control characters */
Xextern int SendType;       /* Which send line to use   */
X
Xextern int Thisbinary;     /* current file is to be received in bin mode */
Xextern int Blklen;         /* record length of received packets */
Xextern char Lzmanag;       /* Local file management request */
Xextern char zconv;         /* ZMODEM file conversion request */
Xextern char zmanag;        /* ZMODEM file management request */
Xextern char ztrans;        /* ZMODEM file transport request */
Xextern jmp_buf tohere;     /* For the interrupt on RX timeout */
Xextern jmp_buf busjmp;     /* for bus errors */
Xextern jmp_buf addrjmp;         /* for address errors */
Xextern unsigned long BusErr, AddrErr;    /* saved vector addresses */
Xextern int buserr();       /* Bus error handler */
Xextern int addrerr();      /* address error handler */
```

```
Xextern int Modem;     /* Send using Xmodem */
Xextern FILE *logf;
Xextern FILE *STDERR;
Xextern int vdebug;
Xextern int lsct;
Xextern int tryzhdrtype;        /* Header type to send corresponding to Last rx
close */
Xextern int Txfcs32;        /* TRUE means send binary frames with 32 bit FCS */
X
X    /* Globals used by ZMODEM functions */
Xextern int Rxframeind;     /* ZBIN or ZHEX indicates type of frame received */
Xextern int Rxtype;         /* Type of header received */
Xextern int Rxcount;        /* Count of data bytes received */
Xextern int Rxtimeout;      /* Tenths of seconds to wait for something */
Xextern char Rxhdr[];       /* Received header */
Xextern char Txhdr[];       /* Transmitted header */
Xextern long Rxpos;         /* Received file position */
Xextern long Txpos;         /* Transmitted file position */
Xextern char Attn[];        /* Attention string rx sends to tx on err */
X
X    /* Globals specific to Sz */
Xextern int Baudrate;
Xextern int Ascii;     /* Add CR's for brain damaged programs */
Xextern int Fullname;       /* transmit full pathname */
Xextern int Unlinkafter;         /* Unlink file after it is sent */
Xextern int Dottoslash;     /* Change foo.bar.baz to foo/bar/baz */
Xextern int errcnt;         /* number of files unreadable */
Xextern int Optiong;        /* Let it rip no wait for sector ACK's */
SHAR_EOF
echo "End of part 1"
echo "File COMMON.H is continued in part 2"
echo "2" > s2_seq_.tmp
exit 0
```


From -9047128125025693054
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,e5083731132061a0,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-29 12:06:27 PST
Path: utzoo!utgpu!water!watmath!clyde!att!osu-cis!tut.cis.ohio-state.edu!mailrus!
nrl-cmf!ames!umd5!purdue!decwrl!ucbvax!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i042: zmdm -- terminal program with file transfer part02/08
Keywords: mwc, alcyon, shar
Message-ID: <3504@saturn.ucsc.edu>
Date: 29 May 88 19:06:27 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1830
Approved: koreth@ssyx.ucsc.edu

Submitted-by: bammi@mandrill.ces.cwru.edu (Jwahar R. Bammi)

```
Posting-number: Volume 1, Issue 42
Archive-name: zmdm/part02

#!/bin/sh
# this is part 2 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file COMMON.H continued
#
CurArch=2
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file COMMON.H"
sed 's/^X//' << 'SHAR_EOF' >> COMMON.H
Xextern int Noeofseen;
Xextern int Totsecs;        /* total number of sectors this file */
Xextern int Filcnt;         /* count of number of files opened */
Xextern int Lfseen;
Xextern unsigned int Rxbuflen;   /* Receiver's max buffer length */
Xextern int Tframlen;       /* Override for tx frame length */
Xextern int blkopt;         /* Override value for zmodem blklen */
Xextern int Rxflags;
Xextern char Lzconv;        /* Local ZMODEM file conversion request */
Xextern char Lztrans;
Xextern int Command;        /* Send a command, then exit. */
Xextern char *Cmdstr;       /* Pointer to the command string */
Xextern int Cmdtries;
Xextern int Cmdack1;        /* Rx ACKs command, then do it */
Xextern int Exitcode;
Xextern int Testattn;       /* Force receiver to send Attn, etc with qbf. */
Xextern char *qbf;
Xextern long Lastread;      /* Beginning offset of last buffer read */
Xextern int Lastc;     /* Count of last buffer read or -1 */
Xextern int Dontread;       /* Don't read the buffer, it's still there */
Xextern jmp_buf intrjmp;         /* For the interrupt on RX CAN */
Xextern jmp_buf abrtjmp;         /* for force abort */
Xextern int siggi;     /* Line interrupt enable flag */
Xextern int Wantfcs32;      /* want to send 32 bit FCS */
Xextern int Znulls;         /* Number of nulls to send at beginning of ZDATA hdr
*/
X
X#ifndef REMOTE
Xextern int *aline_addr;         /* Base addr of aline variables */
X#endif
X
Xextern int hlines;          /* # of lines on screen     */
Xextern int rez;             /* current resolution       */
```

```
Xextern int scolor;             /* current fg/bg screen color toggle */
X
X#ifndef REMOTE
Xextern long *ms_ptr;           /* Pointer to my screen memory */
X#if (MWC || MANX)
Xextern long *m_screen;
X#else
Xextern long m_screen[];
X#endif /* MWC || MANX */
X#endif /* REMOTE */
X
X#ifdef MWC
Xextern long _stksize;
X#endif /* MWC */
X
X#ifdef DLIBS
Xextern long _STKSIZE;
X#endif /* DLIBS */
X
X#ifdef MANX
Xextern long _STKSIZ;
X#endif /* MANX */
X
Xextern struct stat statbuf;           /* Disk Transfer address for Find first etc
*/
Xextern int Baudrate;              /* Current baud rate
               */
Xextern long drv_map;              /* bit vector of valid drives */
X
Xextern BAUDS vbauds[];
Xextern IOREC save,     /* the original Iorec is saved here for the duration
X                of this process */
X    *savep;       /* ptr returned by Iorec() */
X
Xextern char iobuf[]; /* My large Rs232 receive buffer */
X
Xextern char *substr();
X
X#ifdef DYNABUF
Xextern long BBUFSIZ;
Xextern unsigned char *dalloc();
X#endif /* DYNABUF */
X
Xextern long pr_time;
X
X/* -eof- */
SHAR_EOF
echo "File COMMON.H is complete"
chmod 0600 COMMON.H || echo "restore of COMMON.H fails"
echo "x - extracting CONFIG.H (Text)"
sed 's/^X//' << 'SHAR_EOF' > CONFIG.H &&
X/*
X *  Common Configuration file
```

```
X *
X *       Jwahar Bammi
X *            usenet: mandrill!bammi@{decvax,sun}.UUCP
X *            csnet:  bammi@mandrill.ces.CWRU.edu
X *            arpa:   bammi@mandrill.ces.CWRU.edu
X *            CompuServe: 71515,155
X */
X
X
X/*
X *  Compiler Type
X *
X *       Mark Williams C
X *            V2.0 or higher REQUIRED
X *            V3 tested. (V 3.0.6 is what i got)
X *
X *       Alcyon C
X *            V 4.14 only tested
X *            DLIBS compatible
X *            ALN OK
X *
X *       MegaMax C/Laser C (or whatever they call it)
X *            I refuse to use braindamaged tools !!
X *
X *       Manx Aztec C
X *            Version 3.6a tested. Produces great code!
X *
X *       Lattice C
X *            Don't have a copy
X *
X *  Comment out one of the #if 0 lines below corresponding to your
X *  compiler.
X */
X
X    /************** if using Mark Williams C **************/
X
X/* #if 0 */  /* Comment Out this line if using Mark Williams C */
X
X#define COMPILER  "Compiled with Mark Williams C V3.0.6"
X#ifndef MWC
X#define MWC 1
X#endif /* MWC */
X#ifdef DLIBS
X#undef DLIBS      /* only Alcyon version of DLIBS tested */
X#endif /* DLIBS */
X#ifdef ALCYON
X#undef ALCYON
X#endif /* ALCYON */
X#ifdef MANX
X#undef MANX
X#endif /* MANX */
X
X/* #endif */  /* Comment Out this line if using Mark Williams C */
```

```
X
X    /************** if using Alcyon C **************/
X
X#if 0   /* comment out this line if using Alcyon C */
X
X#define COMPILER  "Compiled with Alcyon C V4.14"
X#ifndef ALCYON
X#define ALCYON 1
X#endif /* ALCYON */
X#ifdef MWC
X#undef MWC
X#endif /* MWC */
X#ifdef MANX
X#undef MANX
X#endif /* MANX */
X
X#endif  /* Comment Out this line if using Alcyon C */
X
X    /************** if using MANX C **************/
X
X#if 0   /* comment out this line if using MANX C */
X
X#define COMPILER  "Compiled with Manx Aztec C V3.6a"
X#ifndef MANX
X#define MANX 1
X#endif /* MANX */
X#ifdef MWC
X#undef MWC
X#endif /* MWC */
X#ifdef ALCYON
X#undef ALCYON
X#endif /* ALCYON */
X
X#endif  /* Comment Out this line if using MANX C */
X
X/* ------------------------------------------------------------------------ */
X
X/*
X * Compile Stand Alone  versions of RZ and SZ
X *
X *  STANDALONE is #define'd, to compile `stand alone' alone versions
X *  of RZ and SZ instead of the integrated ZMDM. A lot of folks
X *  requested this feature.
X */
X
X#ifndef STANDALONE
X/* #define STANDALONE 1 */ /* define for standalone compilation */
X#endif /* STANDALONE */
X
X/* ------------------------------------------------------------------------ */
X
X/*
X * Compile a REMOTE version of ZMDM
```

```
X *  REMOTE is #define'd to compile a `remote' version of ZMDM. A remote
X *  versions talks through the serial port, instead of the keyboard/screen
X *  Useful for dialing up your ST to download/upload files. One of the
X *  local BBS's has this version available through a door to enable
X *  people to Up/Down load batch using Zmodem protocol.
X *
X */
X#ifndef REMOTE
X/* #define REMOTE 1 */     /* define for remote version */
X#endif /* REMOTE */
X/* ---------------------------------------------------------------------- */
X
X/*
X * Phone Services
X *  PHONES is #define'd to compile in the Phone Services module.
X *  if PHONES is #define'd you must also define PREDIAL and
X *  REDIAL, your modems dial and re-dial commands respectively.
X *  NOTE: The REMOTE version of ZMDM does not support the
X *  phones module.
X */
X
X#ifndef REMOTE
X/* #if 0 */       /* comment this line if you want the PHONES module */
X
X#define PHONES 1
X    /*********** CAUTION: modem specific ***********/
X#define PREDIAL        "ATDT"    /* Modems dial command */
X#define REDIAL         "A/" /* Modems Re-dial command */
X
X/* #endif */      /* comment this line if you want the PHONES module */
X#endif /* REMOTE */
X
X/* ---------------------------------------------------------------------- */
X
X/*
X * Do flow control while doing terminal emulation -
X * this only needs to be defined if you are
X * going to be running Zmdm at 19200 Baud AND
X * you are going to be sending it (from the host)
X * greater that 16k characters in one blast, without
X * any pause whatsoever. For 99.95 % of us mortals
X * this will never be required. Even if it is defined
X * its no big deal, as flow control will not happen
X * unless the condition described above exists.
X * NOTE: flow control is turned off during file transfers.
X */
X/* #define FLOW_CTRL  1 */ /* do flow control */
X
X/* ----------------------------------------------------------------------*/
X
X/*
X *  DYNABUF
X *  If #define'd use up rest up memory less LEAVEALONE bytes for
```

```
X *  file buffers. The minimum acceptable size is MINACC
X */
X/* #if 0 */  /* Comment this line to use DYNABUF */
X
X#ifndef DYNABUF
X#define DYNABUF 1
X#define LEAVEALONE 16384L
X#define MINACC       8096L
X#endif
X
X/* #endif */ /* Comment this line to use DYNABUF */
X
X/* ------------------------------------------------------------------------*/
X
X/*
X * Size of file buffer
X *
X *  Must be defined. Must be a long. (only define when
X *  DYNABUF is NOT define'd).
X *
X */
X#ifndef DYNABUF
X#define BBUFSIZ       32768L        /* Size of file/capture buffer */
X#endif /* DYNABUF */
X
X/* ------------------------------------------------------------------------*/
X
X/*
X * Size of Rs232 buffer
X *
X *  Must be defined. Must be int (16 bits).
X */
X#define IBUFSIZ       16*1024   /* Size of my Rs232 receive buffer */
X
X/* ------------------------------------------------------------------------*/
X
X/*
X * GMTDIFF
X *  Number of seconds from GMT (signed).
X *  Must be defined.
X *  Must be long.
X */
X#define GMTDIFF   (-5L*3600L)   /* EST (usa) -ve means behind GMT */
X
X/* ------------------------------------------------------------------------*/
X
X/*
X * Define RECURSE only if you want the expand a directory
X * to all its children feature for `sz', when you specify
X * a directory name as an arg to `sz'.
X *
X */
X#define RECURSE 1
```

```
X
X/* ---------------------------------------------------------------------------*/
X
X/*
X * define BIGSTACK if you have very deep directory hierarchies
X *
X */
X#ifdef RECURSE
X/* #define BIGSTACK 1 */
X#endif /* RECURSE */
X
X/* ---------------------------------------------------------------------------*/
X
X
X/* -eof- */
SHAR_EOF
chmod 0600 CONFIG.H || echo "restore of CONFIG.H fails"
echo "x - extracting DECL.H (Text)"
sed 's/^X//' << 'SHAR_EOF' > DECL.H &&
X    /*********************************************\
X    *                             *
X    *  Decl.h - extern declarations for C libarary  *
X    *      must come after <stdio.h> is included    *
X    *                             *
X    *    Preprocessor symbol `MWC' should be        *
X    *    defined when using Mark Williams C       *
X    *                             *
X    *    Preprocessor symbol `MANX' should be       *
X    *    defined when using Manx Aztec C      *
X    *                             *
X    \*********************************************/
X
X#ifndef DECL_H
X
Xextern  FILE *fopen(), *freopen(), *fdopen();
X#if (!(MANX || MWC))
X#ifndef DLIBS
Xextern  FILE *fopena(), *freopa();
Xextern  FILE *fopenb(), *freopb();
X#endif /* DLIBS */
X#endif /* MANX || MWC */
X
X#if MANX
Xextern FILE *tmpfile();
X#endif
X
Xextern  char *etoa();
Xextern  char *ftoa();
X#if (!(MWC || MANX))
X#ifndef DLIBS
Xextern  char *getpass();
X#endif
X#endif
```

```
X
X#ifndef MANX
Xextern  char *index();
Xextern  char *rindex();
X#endif
X
Xextern  char *mktemp();
Xextern  char *strcat();
Xextern  char *strcpy();
Xextern  char *strncat();
Xextern  char *strncpy();
Xextern  char *calloc(), *malloc(), *realloc();
X#if (MWC || MANX || DLIBS)
Xextern char *getenv();
X#endif
X
X#if (!(MWC || MANX))
Xextern  char *sbrk();
X#endif
Xextern  char     *gets(), *fgets();
X#if (!(MWC || MANX))
X#ifndef DLIBS
Xextern  char     *ttyname();
X#endif
X#endif
X
X#if (MWC || MANX)
Xextern  char    *lmalloc(), *lcalloc(), *lrealloc();
X#endif
X
Xextern  double   atan();
Xextern  double   atof();
Xextern  double   ceil();
Xextern  double   cos();
Xextern  double   exp();
Xextern  double   fabs();
Xextern  double   floor();
Xextern  double   fmod();
Xextern  double   log();
Xextern  double   pow();
Xextern  double   sin();
Xextern  double   sinh();
Xextern  double   sqrt();
Xextern  double   tan();
Xextern  double   tanh();
X
Xextern  int  strlen();
Xextern  int  (*signal())();
X
Xextern  long     atol();
Xextern  long     ftell();
Xextern  long getl();
X
```

```
X#ifndef DLIBS
Xextern  long     lseek(), tell();
X#endif
X
X#if (MWC || MANX)
Xextern char  *memchr(), *memcpy(), *memset();
Xextern char  *strchr(), *strerror(), *strpbrk(), *strrchr();
Xextern char  *strstr(), *strtok();
Xextern double     log10();
Xextern double     frexp();
Xextern double     ldexp();
Xextern double     modf();
Xextern double     asin(), acos(), atan2(), cosh();
X#endif /* MWC || MANX */
X
X#ifdef MANX
Xextern char  *memmove(), *memccpy(), *lmemccpy(), *lmemcpy();
Xextern char  *lmemmove(), *lmemchr(), *lmemset(), *strdup();
Xextern char  *scdir(), *tmpnam(), *bsearch();
Xextern long  labs();
Xextern double     cotan();
X
X#define index     strchr
X#define rindex    strrchr
X
X#endif /* MANX */
X
X#define DECL_H
X#endif
X
X/* -eof- */
SHAR_EOF
chmod 0600 DECL.H || echo "restore of DECL.H fails"
echo "x - extracting EXPANDAR.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > EXPANDAR.C &&
X
X    /*
X     *   Examine each argument given to expandargs()
X     *   If it is a directory, then expand it
X     *   to all its component files, recursively
X     *   till you bottom out.
X     *   If it is not a directory, then just pass it on.
X     *
X     *   Inputs: routine, argc, argv
X     *   Outputs: nargc, nargv (calls routine(nargc, nargv))
X     *   To test: compile with -DTEST
X     *         define MWC if using Mark Williams C
X     *         run with a directory as an arg
X     *   Author: JRB   bammi@mandrill.ces.CWRU.edu
X     *   Requirements: Mark Williams C or Alcyon C
X     *        Wants lots of Dynamic memory. It
X     *        all depends upon how many files you
X     *        have. Use the -P option to prune
```

```
X     *       out subdirectories and do things
X     *       one at a time, if you keep running
X     *       out of memory.
X     *       With Mark Williams i use _stksize = 128K
X     *       With Alcyon i use memory model 2 (half of
X     *       of avail memory) in GEMSTART.S
X     *
X          *      WARNINGS: Be CAREFUL about the 40 folder bug. Use
X     *             GEMBOOT or FOLDRXXX when dealing with
X     *           a deeply nested file structure.
X     *
X     *   Added -P name prune option at the suggestion of dietz@zhmti
X     *   Multiple -P's may be given on the command line.
X     *   -P name may be given anywhere on the command line. -P name
X     *   will prune the subdirectories named as arguement to -P.
X     *   ie: when the program is decending the file hierarchy it
X     *   will not visit the pruned branches.
X     *   NOTE that the option is -P and not -p
X     *         (-p is a valid sz option).
X     *   NOTE that the arguement given to -P can be the name
X     *   of a file or a directory. In case it is a file,
X     *   that file is skipped.
X     *
X     */
X
X/*
X * Expand argc, so that the called routine(nargc,nargv) receives only
X * filenames, in nargv[][]
X *
X ************************************************************************
X *                                          *
X *      WARNING:  Be CAREFUL about the 40 folder bug. Use
          *
X *           GEMBOOT or FOLDRXXX when dealing with
          *
X *        a deeply nested file structure.             *
X *                                          *
X ************************************************************************
X *
X *  Jwahar R. Bammi
X *           usenet: mandrill!bammi@{decvax,sun}.UUCP
X *           csnet:  bammi@mandrill.ces.CWRU.edu
X *           arpa:   bammi@mandrill.ces.CWRU.edu
X *           CompuServe: 71515,155
X *
X */
X
X#ifdef TEST
X#include <stdio.h>
X#include <osbind.h>
X#include <ctype.h>
X#endif
X
```

```
X#ifdef TRUE
X#undef TRUE
X#endif
X#ifdef OK
X#undef OK
X#endif
X#ifdef FALSE
X#undef FALSE
X#endif
X
X#define TRUE  1
X#define OK    0
X#define FALSE     0
X#define Realloc   realloc
X
X#ifdef TEST
Xstruct  stat
X{
X    char st_sp1[21];    /* Junk    */
X    char st_mode;       /* File attributes */
X    int  st_time;       /* Mod Time    */
X    int     st_date;       /* Mod date     */
X    long st_size;       /* File size    */
X    char st_name[14];   /* File name    */
X};
X#endif
X
Xtypedef struct _prunelist {
X    char *name;             /* name of subdirectory to prune */
X    struct _prunelist *next;    /* ptr to next */
X} PRUNELIST;
X
Xstatic char *ProgName;
Xstatic PRUNELIST *PruneList = (PRUNELIST *)NULL; /* Head of PruneList */
X
Xstatic char **CopyToNargv();
Xstatic char **ExpandStack();
Xstatic void FreeStack();
Xstatic void FreeNargv();
Xstatic void FreePrune();
Xstatic void FreeUp();
Xstatic int PushDir();
Xstatic char *PopDir();
Xstatic int ProcessDirs();
Xstatic PRUNELIST *AddPrune();
Xstatic int OnPruneList();
Xstatic int isdir();
X
Xextern int existd();
X
Xexpandargs(routine, argc, argv)
Xint (*routine)();
Xint argc;
```

```
Xchar **argv;
X{
X    register int status;
X    int nargc;
X    char **nargv;
X    extern char **CopyToNargv();
X    extern PRUNELIST *AddPrune();
X    extern int existd();
X
X    nargc = 0;
X    nargv = (char **)NULL;
X    ProgName = *argv;
X
X    /* copy argv[0] blindly */
X    if((nargv = CopyToNargv(*argv, nargc, nargv)) == (char **)NULL)
X    {
X        FreeUp(nargc, nargv);
X        return(~OK);
X    }
X
X    nargc++;
X
X    while((--argc) > 0)
X    {
X         argv++;
X         if(**argv == '-')
X         {
X              /* copy any options except -P */
X#ifdef TEST
X              /* some shell pass -P as -p */
X              if( ((*argv)[1] == 'P') || ((*argv)[1] == 'p'))
X#else
X              if( (*argv)[1] == 'P')
X#endif
X              {
X                   if((--argc) <= 0)
X                   {
X                        fprintf(STDERR,"no argument given to -P\n");
X                        FreeUp(nargc, nargv);
X                        return(~OK);
X                   }
X                   if((PruneList = AddPrune(PruneList,*++argv))
X                      == (PRUNELIST *)NULL)
X                   {
X                       FreeUp(nargc, nargv);
X                       return(~OK);
X                   }
X              }
X              else
X              {
X                   if((nargv = CopyToNargv(*argv, nargc, nargv))
X                      == (char **)NULL)
X                   {
```

```
X                    FreeUp(nargc, nargv);
X                    return(~OK);
X                }
X                else
X                    nargc++;
X            }
X
X        }
X        else
X        {
X            /* If its not on the PruneList then */
X            if(!OnPruneList(PruneList, *argv))
X            {
X                /* if it is a directory, push it */
X                if(existd(*argv))
X                {
X                    if((status = PushDir(*argv)) != OK)
X                    {
X                        FreeUp(nargc, nargv);
X                        return(status);
X                    }
X                }
X                else
X                {
X                    /* it is NOT a directory, copy to nargv */
X                    if((nargv = CopyToNargv(*argv, nargc, nargv))
X                       == (char **)NULL)
X                    {
X                        FreeUp(nargc, nargv);
X                        return(~OK);
X                    }
X                    else
X                        nargc++;
X                }
X            }
X        }
X    } /* while */
X
X    /* process pushed directories if any */
X    if((status = ProcessDirs(&nargc, &nargv)) != OK)
X    {
X        FreeUp(nargc, nargv);
X        return(status);
X    }
X    /* else Free the Stack and Prune List, call *routine */
X    FreeStack();
X    FreePrune();
X    status =  (*routine)(nargc, nargv);
X    FreeNargv(nargc, nargv);
X
X    return status;
X
X}
```

```
X
X/*
X * Expand nargv by an element and copy a String into the new element
X *
X */
Xstatic char **CopyToNargv(string, nargc, nargv)
Xchar *string;
Xint nargc;
Xchar **nargv;
X{
X    extern char *malloc(), *Realloc(), *strcpy();
X    extern int strlen();
X
X    /* expand nargv by 1 element */
X    if(nargv == (char **)NULL)
X        /* do it with malloc for the first one */
X        nargv = (char **)malloc(sizeof(char **));
X    else
X        /* do it with Realloc for others */
X        nargv = (char **)Realloc(nargv, (nargc+1)*sizeof(char **));
X
X    if(nargv == (char **)NULL)
X    {
X         /* failed to get memory */
X         fprintf(STDERR,"%s(CopyToNargv()): Out of Memory\n", ProgName);
X         return ((char **)NULL);
X    }
X
X    /* Get mem for string */
X    if(( nargv[nargc] = malloc(strlen(string)+1)) == (char *)NULL)
X    {
X         /* failed to get memory */
X         fprintf(STDERR,"%s(CopyToNargv()): Out of Memory\n", ProgName);
X         return ((char **)NULL);
X    }
X
X    /* copy string into nargv[nargc] */
X    (void)strcpy( nargv[nargc], string);
X    return(nargv);
X}
X
Xstatic char **Stack = (char **)NULL; /* directory stack */
Xstatic char StackSize = 0;      /* Size of current Stack */
Xstatic int    Top   = -1;
X#define STACK_EMPTY   (Top < 0)
X#define CHUNKSIZE 16
X
X/*
X * Grow the Stack by one chunk of CHUNKSIZE elements
X *
X */
Xstatic char **ExpandStack(Stack)
Xchar **Stack;
```

```
X{
X    extern char *malloc(), *Realloc();
X
X    /* Grow Stack */
X    if(Stack == (char **)NULL)
X        /* with malloc */
X        Stack = (char **)malloc(CHUNKSIZE * sizeof(char **));
X    else
X        /* with Realloc */
X        Stack = (char **)Realloc(Stack, (StackSize+CHUNKSIZE) *
X                                 sizeof(char **));
X
X    if(Stack == (char **)NULL)
X    {
X         /* outa mem */
X         fprintf(STDERR,"%s(ExpandStack()): Out of Memory\n", ProgName);
X         return((char **)NULL);
X    }
X    StackSize += CHUNKSIZE;
X    return(Stack);
X}
X
X/*
X * Free the Stack
X *
X */
Xstatic void FreeStack()
X{
X    if(StackSize > 0)
X         (void)free(Stack);
X    Stack = (char **)NULL;
X    StackSize = 0;
X    Top   = -1;
X}
X
X/*
X * Free Nargv
X *
X */
Xstatic void FreeNargv(nargc, nargv)
Xint nargc;
Xchar *nargv[];
X{
X    register int i;
X
X    for(i = 0; i < nargc; i++)
X         (void)free(nargv[i]);
X    if(nargc > 0)
X         (void)free(nargv);
X}
X
X/*
X * Free the PruneList
```

```
X *
X */
Xstatic void FreePrune()
X{
X    register PRUNELIST *p, *next;
X
X    for(p = PruneList; p != (PRUNELIST *)NULL; p = next)
X    {
X         next = p->next;
X         (void)free(p);
X    }
X    PruneList = (PRUNELIST *)NULL;
X}
X
X/*
X * FreeUp before bad exit
X *
X */
Xstatic void FreeUp(nargc, nargv)
Xint nargc;
Xchar **nargv;
X{
X    FreeStack();
X    FreePrune();
X    FreeNargv(nargc, nargv);
X}
X
X/*
X * Push a directory name on Stack
X *
X */
Xstatic int PushDir(name)
Xchar *name;
X{
X    extern char *malloc(), *strcpy();
X    extern int strlen();
X    extern char **ExpandStack();
X
X#ifdef DDEBUG
Xprintf("PushDir: %s\n", name);
X#endif
X
X    ++Top;
X    if(Top >= StackSize)
X    {
X         if((Stack = ExpandStack(Stack)) == (char **)NULL)
X             return(~OK);
X    }
X
X    if((Stack[Top] = malloc(strlen(name)+1)) == (char *)NULL)
X    {
X         /* outa mem */
X         fprintf(STDERR,"%s(PushDir()): Out of Memory\n", ProgName);
```

```
X         return(~OK);
X    }
X    (void)strcpy(Stack[Top], name);
X    return(OK);
X}
X
X/*
X * Pop a directory name from the stack
X *
X */
Xstatic char *PopDir()
X{
X    register char *r;
X    extern char **ShrinkStack();
X
X    if(STACK_EMPTY)
X        return ((char *)NULL);
X
X    r = Stack[Top];
X    Top--;
X    return(r);
X}
X
Xstatic int BadStatus = FALSE;
X#define BADSTATUS (BadStatus != FALSE)
X#define MAXNAMLEN 128
X
X/*
X * Process directories on the Stack, by adding all the
X * files in a directory to nargv.
X */
Xstatic int ProcessDirs(nargc, nargv)
Xint *nargc;
Xchar ***nargv;
X{
X    register char *name;
X    register struct stat *dp;
X    register int status, slashp;
X    char path[MAXNAMLEN+1];
X    extern char **CopyToNargv();
X    extern char *PopDir();
X    extern char *alltolower();
X
X    if(BADSTATUS)
X        return(~OK);
X
X    if(STACK_EMPTY)
X        /* Nothing more to do */
X        return(OK);
X
X    /* Pop a directory from Stack and process */
X    if((name = PopDir()) == (char *)NULL)
X    {
```

```
X         /* Oh Oh */
X         fprintf(STDERR,"Internal Error (BUG), PopDir returns NULL\n");
X         BadStatus = (~FALSE);
X         return(~OK);
X    }
X
X    strcpy(path, name);
X    if(path[(strlen(path)-1)] == '\\')
X    {
X         strcat(path,"*.*");
X         slashp = TRUE;
X    }
X    else
X    {
X         strcat(path,"\\*.*");
X         slashp = FALSE;
X    }
X
X    /* Open the directory */
X    if(Fsfirst(path, 0x0020| 0x0010 | 0x0001) != 0)
X    {
X         /* trouble opening directory */
X         fprintf(STDERR,"Trouble opening %s\n",path);
X         /* set BADSTATUS and return */
X         BadStatus = (~FALSE);
X         return(~OK);
X    }
X
X    /* get the DTA */
X    dp = (struct stat *)Fgetdta();
X
X    /* for each entry in the directory, if it is a file
X       add to nargv. If it is a directory, Push it onto
X       the directory stack.
X     */
X    do
X    {
X         if(! ((strcmp(dp->st_name,".") == 0) ||
X               (strcmp(dp->st_name,"..") == 0)) )
X         {
X              strcpy(path, name);
X              if(!slashp)
X                   strcat(path,"\\");
X              strcat(path,dp->st_name);
X
X              /* If this path is on the PruneList skip */
X              if(OnPruneList(PruneList, alltolower(path)))
X                  continue;
X
X              if(!isdir(path, dp->st_mode))
X              {
X                   /* not a dir -- add this to nargv */
X                   if((*nargv = CopyToNargv(path, *nargc, *nargv))
```

```
X                       == (char **)NULL)
X                   {
X                        BadStatus = (~FALSE);
X                        return(~OK);
X                   }
X                   else
X                   {
X                        *nargc += 1;
X                   }
X
X              }
X              else
X              {
X
X                   /* Push This directory */
X                   if((status = PushDir(path)) != OK)
X                   {
X                        BadStatus = (~FALSE);
X                        return(status);
X                   }
X              }
X         }
X    } while(Fsnext() == 0);
X
X    free(name);   /* done with this directory */
X
X    /* go do the rest */
X    return (ProcessDirs(nargc, nargv));
X}
X
X/*
X * Add a name to PruneList
X *
X */
Xstatic PRUNELIST *AddPrune(list, name)
XPRUNELIST *list;
Xchar *name;
X{
X    extern char *malloc();
X    register PRUNELIST *new;
X
X    if((new = (PRUNELIST *)malloc(sizeof(PRUNELIST))) == (PRUNELIST *)NULL)
X    {
X         /* outa mem */
X         fprintf(STDERR,"%s(AddPrune()): Out of Memory\n", ProgName);
X         return(new);
X    }
X
X    new->name = name;
X    new->next = list;
X
X    return(new);
X}
```

```
X
X/*
X * Search for name on PruneList
X *
X */
Xstatic int OnPruneList(list, name)
Xregister PRUNELIST *list;
Xregister char *name;
X{
X    for(; list != (PRUNELIST *)NULL; list = list->next)
X    {
X         if(strcmp(list->name, name) == 0)
X             return(~FALSE);
X    }
X
X    return(FALSE);
X}
X
X/*
X * test if a subdirectory exists, without touching the DTA
X * include special case of 'D:\' that Fsfirst does'nt handle correctly
X */
Xstatic int isdir(name, attr)
Xregister char *name;
Xregister int attr;
X{
X    /* assuming the DTA buffer is already set up */
X    extern long drv_map;
X    register int drive;
X
X    if(attr & 0x0010)
X         return TRUE;
X
X    /* Gemdos doesn't like d:\ style dirs */
X    if((name[3] == '\0') && (name[2] == '\\') && (name[1] == ':'))
X    {
X         drive = name[0];
X         if(isupper(drive))
X              drive = tolower(drive);
X
X         drive = drive - 'a';
X         if((drv_map & (1L << drive)) == 0)
X              return FALSE;
X         else
X              return TRUE;
X    }
X    /* Nor does Gemdos understand '.' or '..' */
X    /* Hey Atari, don't you guys ever test anything */
X    if((strcmp(name,".") == 0) || (strcmp(name,"..") == 0) ||
X       (strcmp(name,".\\") == 0) || (strcmp(name,"..\\") == 0))
X         return TRUE;
X
X    return FALSE;
```

```
X}
X
X
X#ifdef TEST
X
X/*
X * convert string to all lower case
X */
Xchar *alltolower(s)
Xchar *s;
X{
X    register char *p;
X
X    for(p = s; *p != '\0'; p++)
X         if(isupper(*p))
X              *p = tolower(*p);
X
X    return s;
X}
X
X/*
X * test if a subdirectory exists
X * include special case of 'D:\' that Fsfirst does'nt handle correctly
X * (this routine not needed for zmdm, as it is defined in common.c)
X */
Xint existd(name)
Xregister char *name;
X{
X    /* assuming the DTA buffer is already set up */
X    /* assumes drv_map has been read in drv_map externally */
X    extern long drv_map;
X    register int drive;
X    extern struct stat statbuf;
X
X    if (Fsfirst(name , 0x0021|0x0010) == 0)
X    {
X         if((statbuf.st_mode & 0x0010) == 0x0010)
X              return TRUE;
X    }
X
X    /* Gemdos doesn't like d:\ style dirs */
X    if((name[3] == '\0') && (name[2] == '\\') && (name[1] == ':'))
X    {
X         drive = name[0];
X         if(isupper(drive))
X              drive = tolower(drive);
X
X         drive = drive - 'a';
X         if((drv_map & (1L << drive)) == 0)
X              return FALSE;
X         else
X              return TRUE;
X    }
```

```
X    /* Nor does Gemdos understand '.' or '..' */
X    /* Hey Atari, don't you guys ever test anything */
X    if((strcmp(name,".") == 0) || (strcmp(name,"..") == 0) ||
X       (strcmp(name,".\\") == 0) || (strcmp(name,"..\\") == 0))
X         return TRUE;
X
X    return FALSE;
X}
X
Xint MAIN(argc, argv)
Xint argc;
Xchar **argv;
X{
X    register int i;
X
X    for(i = 0; i < argc; i++)
X        printf("%d:\t%s\n", i, argv[i]);
X
X
X    return(0);
X}
X
X
Xstruct stat statbuf;    /* Disk Transfer address for Find first etc */
Xlong drv_map;
X
X#ifdef MWC
Xlong _stksize = 128L * 1024L;
X#endif
X
X#ifdef MANX
Xlong _STKSIZ = 128L * 1024L;
X#endif
X
Xmain(argc, argv)
Xint argc;
Xchar **argv;
X{
X    /* Set up Dta */
X    Fsetdta(&statbuf);
X    drv_map = Drvmap();
X
X    exit(expandargs(MAIN, argc, argv));
X}
X
X#endif /* TEST */
X
X/* -eof- */
SHAR_EOF
chmod 0600 EXPANDAR.C || echo "restore of EXPANDAR.C fails"
echo "x - extracting FILEIO.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > FILEIO.C &&
X/*
```

```
X *  File I/O (with large buffers) Module
X *
X *      Jwahar Bammi
X *           usenet: mandrill!bammi@{decvax,sun}.UUCP
X *           csnet:  bammi@mandrill.ces.CWRU.edu
X *           arpa:   bammi@mandrill.ces.CWRU.edu
X *           CompuServe: 71515,155
X */
X
X#include "config.h"
X
X#include "zmdm.h"
X#include "common.h"
X
X#define  O_RDONLY  1
X#define O_WRONLY  2
X#define O_APONLY  4
X#ifndef DYNABUF
X#define MBUFSIZ       (((long)BBUFSIZ)-1L)
X#else
Xstatic long MBUFSIZ;
X#endif /* DYNABUF */
X
Xstatic unsigned char *bptr;
X
Xstatic long flcount = (-1L);
Xstatic int bufmode;
X
Xint stfopen(name, mode)
Xchar *name, *mode;
X{
X    register int handl;
X
X    switch(*mode)
X    {
X        case 'r':
X         if((handl = Fopen(name, 0)) <= 0)
X              return -1;
X         bufmode = O_RDONLY;
X         break;
X
X        case 'w':
X         if((handl = Fcreate(name, 0)) <= 0)
X         {
X              if((handl = Fopen(name, 1)) <= 0)
X                   return -1;
X         }
X         bufmode = O_WRONLY;
X         break;
X
X        case 'a':
X         if((handl = Fopen(name, 2)) <= 0)
X              return -1;
```

```
X         Fseek(0L, handl, 2);
X         bufmode = O_APONLY;
X         break;
X
X       default:
X         return -1;
X    }
X
X#ifdef DYNABUF
X    MBUFSIZ = BBUFSIZ - 1L;
X#endif /* DYNABUF */
X
X    bptr = bufr;
X    flcount = (-1L);
X    return handl;
X}
X
Xstfclose(handl)
Xint handl;
X{
X    if(bufmode == O_RDONLY)
X         return Fclose(handl);
X    if(stflush(handl))
X    {
X         Fclose(handl);
X         return -1;
X    }
X    return Fclose(handl);
X}
X
Xstputc(c, handl)
Xunsigned int c;
Xint handl;
X{
X    if(flcount >= MBUFSIZ)
X    {
X         if(Fwrite(handl, (long)BBUFSIZ, bufr) != (long)BBUFSIZ)
X              return -1;
X         flcount = (-1L);
X         bptr  = bufr;
X    }
X    flcount++;
X    *bptr++ = c;
X    return 0;
X}
X
Xstgetc(handl)
Xint handl;
X{
X    if(flcount <= 0)
X    {
X         if((flcount = Fread(handl, (long)BBUFSIZ, bufr)) == 0)
X              return EOF;
```

```
X         bptr = bufr;
X    }
X    flcount--;
X    return(*bptr++);
X}
X
Xstflush(handl)
Xint handl;
X{
X    if(flcount < 0)
X         return 0;
X
X    if(Fwrite(handl, (long)(flcount+1L), bufr) != (flcount+1L))
X         return -1;
X    flcount = (-1L);
X    bptr = bufr;
X    return 0;
X}
X
Xstfseek(handl, disp, mode)
Xint handl;
Xlong disp;
Xint  mode;
X{
X    if(bufmode != O_RDONLY)
X         if(stflush(handl))
X              return -1;
X    Fseek(disp, handl, mode);
X    flcount = (-1L);
X    bptr = bufr;
X    return 0;
X}
X
X
X/* -eof- */
SHAR_EOF
chmod 0600 FILEIO.C || echo "restore of FILEIO.C fails"
echo "x - extracting HI5025.S (Text)"
sed 's/^X//' << 'SHAR_EOF' > HI5025.S &&
X/
X/  Make hi rez screen bios handle 50 lines of 8x8 characters
X/
X/  Adapted to Mark Williams C use from origional PD asm posting
X/  from atari corp.
X/
X/       Jwahar Bammi
X/            usenet: cwruecmp!bammi@decvax.UUCP
X/            csnet:  bammi@cwru.edu
X/            arpa:   bammi@cwru.edu
X/            CompuServe: 71515,155
X/
X/
X         .shri
```

```
X
X    .globl hi50_
X
Xhi50_:                  / switch to 50 line mode
X    link a6,$0          / routine preamble
X
X    .word    0xA000          / get the important pointers (line A init)
X
X    movea.l    4(a1),a1      / a1 -> 8x8 font header
X
X    move.l  72(a1),-0x0A(a0)  / v_off_ad <- 8x8 offset table addr
X    move.l  76(a1),-0x16(a0)  / v_fnt_ad <- 8x8 font data addr
X
X    move    $8,  -0x2E(a0)    / v_cel_ht <- 8    8x8 cell height
X    move    $49, -0x2A(a0)    / v_cel_my <- 49   maximum cell "Y"
X    move    $640,-0x28(a0)    / v_cel_wr <- 640  offset to cell Y+1
X
X    unlk a6         / routine postable
X    rts             / and return
X
X
X/
X/ Make hi rez screen bios handle 25 lines of 8x16 characters
X/
X
X    .globl hi25_
X
Xhi25_:                  / Switch to 25 lines display
X    link a6,$0          / routine preamble
X
X    .word   0xA000           / get the important pointers
X
X    movea.l    8(a1),a1      / a1 -> 8x16 font header
X
X    move.l  72(a1),-0x0A(a0)  / v_off_ad <- 8x16 offset table addr
X    move.l  76(a1),-0x16(a0)  / v_fnt_ad <- 8x16 font data addr
X
X    move    $16,  -0x2E(a0)   / v_cel_ht <- 16    8x16 cell height
X    move    $24,  -0x2A(a0)   / v_cel_my <- 24    maximum cell "Y"
X    move    $1280,-0x28(a0)   /  v_cel_wr <- 1280  vertical byte offset
X
X    unlk a6         / routine postamble
X    rts             / bye
SHAR_EOF
chmod 0600 HI5025.S || echo "restore of HI5025.S fails"
echo "x - extracting LNK (Text)"
sed 's/^X//' << 'SHAR_EOF' > LNK &&
Xc:\lib\gemstart.o common.o rz.o sz.o transfer.o util.o main.o tyme.o
Xzm.o fileio.o phone.o c:\lib\osbind.o c:\lib\gemlib c:\lib\libf
SHAR_EOF
chmod 0600 LNK || echo "restore of LNK fails"
echo "x - extracting MAIN.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > MAIN.C &&
```

```
X/*
X * Main Module
X *
X *      Jwahar Bammi
X *           usenet: mandrill!bammi@{decvax,sun}.UUCP
X *           csnet:  bammi@mandrill.ces.CWRU.edu
X *           arpa:   bammi@mandrill.ces.CWRU.edu
X *           CompuServe: 71515,155
X */
X
X#include "config.h"
X
X#include "zmdm.h"
X#include "common.h"
X
X#ifndef Vsync               /* Atari forgot these in osbind.h */
X#define Vsync()   xbios(37)
X#endif
X
X#define esc  27
X#define cr   0x0d
X#define mvto(r,c) EscSeq('Y');Bconout(2,r+040);Bconout(2,c+040)
X
X
X    /* Globals belonging to this module only */
X
Xint rs232 = 1,         /* Ports */
X        console = 2;
Xint speed,               /* rs232 setup parameters */
X#ifdef FLOW_CTRL
X    flowctl = 1,
X#else
X    flowctl = 0,
X#endif
X    ucr = -1,
X    rsr = -1,
X    tsr = -1,
X    scr = -1;
X
X/*
X * setRs232() - set rs232 port configuration
X */
X#ifndef REMOTE
Xvoid setRs232 ()
X{
X    char ch;
X    long conin;
X
X    Bconws("Baud rate: ");
X    EscSeq('p');
X    Bconws("0=19200 1=9600, 2=4800, 3=2400, 4=1200, 5=300\r\n");
X    EscSeq('q');
X    Bconout(2, '\t');
```

```
X    EscSeq('p');
X    Bconws("What speed==>");
X    EscSeq('q');
X
X    conin = Bconin(console);    /* get speed */
X    if ((conin & 0x00FF0000L) == 0x00610000L)
X    {
X         his_screen();
X         ResetIoBuf();
X         finish();
X    }
X    ch = (char) (conin & 0x007f);
X    Bconout(2, ' ');
X    switch (ch)
X    {
X        case '0':      /* 19200 */
X         speed = 0;
X         Bconws("19200");
X         Baudrate = 19200;
X         break;
X
X        case '1':
X         speed = 1;    /* 9600 */
X         Bconws("9600");
X         Baudrate = 9600;
X         break;
X
X        case '2':
X         speed = 2;    /* 4800 */
X         Bconws("4800");
X         Baudrate = 4800;
X         break;
X
X        case '3':
X         speed = 4;    /* 2400 */
X         Bconws("2400");
X         Baudrate = 2400;
X         break;
X
X        case '4':
X         speed = 7;    /* 1200 */
X         Bconws("1200");
X         Baudrate = 1200;
X         break;
X
X        case '5':
X         speed = 9;    /* 300 */
X         Bconws("300");
X         Baudrate = 300;
X         break;
X
X        default:
X         speed = getbaud();
```

```
X         Bconws(BAUD_STRING(speed));
X         Baudrate = BAUD_RATE(speed);
X    }
X    Bconws(" Baud\r\n");
X
X    /* Set new Baud rate */
X
X/*  Txoff(); */
X    Rsconf(speed, flowctl, ucr, rsr, tsr, scr);
X    Vsync(); Vsync();
X/*  Txon(); */
X
X
X}
X#else
Xvoid setRs232 ()
X{
X    char ch;
X    long conin;
X
X    Bauxws("Baud rate: ");
X    Bauxws("0=19200 1=9600, 2=4800, 3=2400, 4=1200, 5=300\r\n");
X    Bconout(1, '\t');
X    Bauxws("What speed==>");
X
X    conin = Bconin(1); /* get speed */
X    if (((int)(conin & 0x007f) & CTRL('U')) == CTRL('U'))
X    {
X         his_screen();
X         ResetIoBuf();
X         finish();
X    }
X    ch = (char) (conin & 0x007f);
X    Bconout(1, ' ');
X    switch (ch)
X    {
X        case '0':      /* 19200 */
X         speed = 0;
X         Bauxws("19200");
X         Baudrate = 19200;
X         break;
X
X        case '1':
X         speed = 1;    /* 9600 */
X         Bauxws("9600");
X         Baudrate = 9600;
X         break;
X
X        case '2':
X         speed = 2;    /* 4800 */
X         Bauxws("4800");
X         Baudrate = 4800;
X         break;
```

```
X
X       case '3':
X        speed = 4;    /* 2400 */
X        Bauxws("2400");
X        Baudrate = 2400;
X        break;
X
X       case '4':
X        speed = 7;    /* 1200 */
X        Bauxws("1200");
X        Baudrate = 1200;
X        break;
X
X       case '5':
X        speed = 9;    /* 300 */
X        Bauxws("300");
X        Baudrate = 300;
X        break;
X
X       default:
X        speed = getbaud();
X        Bauxws(BAUD_STRING(speed));
X        Baudrate = BAUD_RATE(speed);
X   }
X   Bauxws(" Baud\r\n");
X
X   /* Set new Baud rate */
X
X/*  Txoff(); */
X   Rsconf(speed, flowctl, ucr, rsr, tsr, scr);
X   Vsync(); Vsync();
X/*  Txon(); */
X
X
X}
X#endif /* REMOTE */
X
X/*
X * help() - display help info and menu
X */
X#ifndef REMOTE
Xvoid help ()
X{
X   register long conin;
X   register int x;
X   extern char *r_filename();
X
X   my_screen();       /* Switch to my screen memory */
X   EscSeq('v');       /* wrap at end of line */
X   EscSeq('E');       /* clear screen */
X
X   mvto(2,25);
X   EscSeq('p');
```

```
X    Bconws("ZMDM Version ");
X    Bconws(ZMDMVERSION);
X    EscSeq('q');
X    EscSeq('p');
X    x = strlen(COMPILER);
X    x = (80 - x)/2;
X    mvto(3,x);
X    Bconws(COMPILER);
X    EscSeq('q');
X    mvto(5,25);
X    Bconws("ST Enthusiasts @ ces.CWRU.edu\r\n\n");
X
X    /* Put up menu */
X    Bconws("\r\n\t");
X    EscSeq('p');       /* reverse video */
X    Bconws("Undo");
X    EscSeq('q');       /* quit reverse video */
X    Bconws(" to exit.\r\n");
X
X    Bconws("\t");
X    EscSeq('p');       /* reverse video */
X    Bconws("Help");
X    EscSeq('q');       /* quit reverse video */
X    Bconws(" for this message.\r\n");
X
X    Bconws("\t");
X    EscSeq('p');       /* reverse video */
X    Bconws("Escape");
X    EscSeq('q');       /* quit reverse video */
X    Bconws(" to send a break.\r\n");
X
X    Bconws("\t");
X    EscSeq('p');       /* reverse video */
X    Bconws("T or t");
X    EscSeq('q');       /* quit reverse video */
X    Bconws(" to do file transfers and local functions.\r\n");
X
X#ifdef PHONES
X    Bconws("\t");
X    EscSeq('p');       /* reverse video */
X    Bconws("P or p");
X    EscSeq('q');       /* quit reverse video */
X    Bconws(" for Phone services.\r\n");
X#endif
X
X    if(rez == 2)
X    {
X         Bconws("\t");
X         EscSeq('p');       /* reverse video */
X         Bconws("H or h");
X         EscSeq('q');       /* quit reverse video */
X         Bconws(" for Hi Rez Toggle (25/50 Lines).\r\n");
X    }
```

```
X
X    Bconws("\t");
X    EscSeq('p');       /* reverse video */
X    Bconws("I or i");
X    EscSeq('q');       /* quit reverse video */
X    Bconws(" to Invert screen colors.\r\n");
X
X    Bconws("\t");
X    EscSeq('p');       /* reverse video */
X    Bconws("Return");
X    EscSeq('q');       /* quit reverse video */
X    Bconws(" to do nothing.\r\n");
X
X    Bconws("\t");
X    EscSeq('p');       /* reverse video */
X    Bconws("B or b");
X    EscSeq('q');       /* quit reverse video */
X    Bconws(" to set baud rate.     Currently is ");
X    EscSeq('p');
X    Bconws(BAUD_STRING(speed));
X    Bconws(" Baud.\r\n\r\n");
X    EscSeq('q');
X
X    /* get response */
X    conin = Bconin(console);
X
X    if ((conin & 0x00FF0000L) == 0x00610000L)
X    {
X         /* He hit <UNDO> */
X         his_screen();
X         ResetIoBuf();
X         finish();
X    }
X
X    switch((int)(conin & 0x007f))
X    {
X        case 'B':
X        case 'b':
X         /* Set baud rate */
X         setRs232();
X         break;
X
X        case 'T':
X        case 't':
X            EscSeq('E');        /* clear screen */
X
X             /* Set no flow Control */
X#ifdef FLOW_CTRL
X             Rsconf(-1,0,-1,-1,-1,-1);
X             Vsync(); Vsync();
X
X#endif
X         /* Go do transfers */
```

```
X         transfer();
X
X#ifdef FLOW_CTRL
X              /* Flow Control On */
X/*            Txoff(); */
X              Rsconf(-1,1,-1,-1,-1,-1);
X              Vsync(); Vsync();
X/*            Txon(); */
X#endif
X         his_screen();
X         return;
X
X        case '\033':
X         /* Send a break */
X         sendbrk();
X         his_screen(); /* Don't wait for the key hit */
X
X         return;
X
X        case 'i':
X        case 'I':
X         /* Invert screen colors */
X         his_screen();
X         if(scolor == 0)
X         {
X              EscSeq('b');  /* Foreground color 0 */
X              Bconout(2, 0);
X              EscSeq('c');  /* Background color 1 */
X              Bconout(2, 1);
X              scolor = 1;
X         }
X         else
X         {
X              EscSeq('b');  /* Foreground color 1 */
X              Bconout(2, 1);
X              EscSeq('c');  /* Background color 0 */
X              Bconout(2, 0);
X              scolor = 0;
X         }
X         EscSeq('E');       /* Clear the screen */
X         return;
X
X#ifdef PHONES
X        case 'p':
X        case 'P':
X         /* Phone Services */
X         phone();
X         return;
X#endif
X        case 'h':
X        case 'H':
X         /* Hi rez 25/50 toggle */
X         if(rez == 2)
```

```
X        {
X             if(hlines == 25)
X             {
X                  hlines = 50;
X                  hi50();
X             }
X             else
X             {
X                  hlines = 25;
X                  hi25();
X             }
X             his_screen();
X             EscSeq('E');       /* clear screen */
X             return;
X        }
X        /* else fall Through */
X
X       default:
X        Bconws("No Change\r\n");
X    }
X
X    /* Wait for a key hit */
X    hit_key();
X    /* back to terminal screen */
X    his_screen();
X}
X#else
Xvoid help ()
X{
X    register long conin;
X    extern char *r_filename();
X
X    my_screen();       /* Switch to my screen memory */
X
X    Bauxws("\r\n\n");
X    Bauxws("                         RZMDM Version ");
X    Bauxws(ZMDMVERSION);
X    Bauxws("\r\n                              ");
X    Bauxws(COMPILER);
X    Bauxws("\r\n\n                          ST Enthusiasts @ ces.CWRU.edu\r\n\n");
X
X    /* Put up menu */
X    Bauxws("\r\n\t");
X    Bauxws("CTRL-U");
X    Bauxws(" to exit.\r\n");
X
X    Bauxws("\t");
X    Bauxws("CTRL-Z");
X    Bauxws(" for this message.\r\n");
X
X    Bauxws("\t");
X    Bauxws("Escape");
X    Bauxws(" to send a break.\r\n");
```

```
X
X    Bauxws("\t");
X    Bauxws("T or t");
X    Bauxws(" to do file transfers and local functions.\r\n");
X
X    Bauxws("\t");
X    Bauxws("Return");
X    Bauxws(" to do nothing.\r\n");
X
X    Bauxws("\t");
X    Bauxws("B or b");
X    Bauxws(" to set baud rate.     Curently is ");
X    Bauxws(BAUD_STRING(speed));
X    Bauxws(" Baud.\r\n\r\n");
X
X    /* get response */
X    conin = Bconin(1);
X
X    if (((int)(conin & 0x007f) & CTRL('U')) == CTRL('U'))
X    {
X         /* He hit <UNDO> */
X         his_screen();
X         ResetIoBuf();
X         finish();
X    }
X
X    switch((int)(conin & 0x007f))
SHAR_EOF
echo "End of part 2"
echo "File MAIN.C is continued in part 3"
echo "3" > s2_seq_.tmp
exit 0


From -8515071815066794658
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,422d8d45c96bbafb,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-29 12:13:45 PST
Path: utzoo!utgpu!water!watmath!clyde!att!osu-cis!tut.cis.ohio-state.edu!mailrus!
ames!umd5!purdue!decwrl!ucbvax!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i043: zmdm -- terminal program with file transfer  part03/08
Keywords: mwc, alcyon, shar
Message-ID: <3505@saturn.ucsc.edu>
Date: 29 May 88 19:13:45 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1949
Approved: koreth@ssyx.ucsc.edu

Submitted-by: bammi@mandrill.ces.cwru.edu (Jwahar R. Bammi)
Posting-number: Volume 1, Issue 43
```

```
Archive-name: zmdm/part03

#!/bin/sh
# this is part 3 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file MAIN.C continued
#
CurArch=3
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file MAIN.C"
sed 's/^X//' << 'SHAR_EOF' >> MAIN.C
X    {
X        case 'B':
X        case 'b':
X         /* Set baud rate */
X         setRs232();
X         break;
X
X        case 'T':
X        case 't':
X            Bauxws("\r\n\n");        /* clear screen */
X
X             /* Set no flow Control */
X#ifdef FLOW_CTRL
X             Rsconf(-1,0,-1,-1,-1,-1);
X             Vsync(); Vsync();
X
X#endif
X         /* Go do transfers */
X         transfer();
X
X#ifdef FLOW_CTRL
X             /* Flow Control On */
X/*           Txoff(); */
X             Rsconf(-1,1,-1,-1,-1,-1);
X             Vsync(); Vsync();
X/*           Txon(); */
X#endif
X         his_screen();
X         return;
X
X        case '\033':
X         /* Send a break */
X         sendbrk();
X         his_screen(); /* Don't wait for the key hit */
```

```
X
X        return;
X
X        /* else fall Through */
X
X       default:
X        Bauxws("No Change\r\n");
X    }
X
X    /* back to terminal screen */
X    his_screen();
X}
X#endif /* REMOTE */
X
Xmain ()
X{
X    register int  c;       /* rs232 input */
X    register int  i;
X    register long conin;
X#ifndef REMOTE
X    extern int *aaddress(); /* Routine that returns base address of
X                   * line A variables
X                   */
X#else
X    extern FILE *fopen();
X#endif
X
X#ifdef MWC
X    extern char *lmalloc();
X#endif
X
X    /* Set up Dta */
X    Fsetdta(&statbuf);
X
X    /* Get screen rez */
X    rez = Getrez();
X    drv_map = Drvmap();
X
X#if (MWC || MANX)
X#ifndef REMOTE
X#ifdef MWC
X       if((m_screen = (long *)lmalloc(
X                   (unsigned long)((8L*1024L+32L)*(long)sizeof(long))))
X                                == (long *)NULL)
X#else
X       if((m_screen = (long *)Malloc(
X                   (unsigned long)((8L*1024L+32L)*(long)sizeof(long))))
X                                == (long *)NULL)
X#endif
X    {
X        Bconws("Sorry, could not allocate enough memory\r\n");
X        Pterm(3);
X    }
```

```
X#endif /* REMOTE */
X#ifndef DYNABUF
X#ifdef MANX
X    if((bufr = (unsigned char *)Malloc((unsigned long)BBUFSIZ))
X                        == (unsigned char *)NULL)
X#else
X    if((bufr = (unsigned char *)lmalloc((unsigned long)BBUFSIZ))
X                        == (unsigned char *)NULL)
X#endif
X#else
X    if((bufr = dalloc()) == (unsigned char *)NULL)
X#endif /* DYNABUF */
X    {
X#ifdef REMOTE
X        Bauxws("Sorry, could not allocate enough memory\r\n");
X#else
X        Bconws("Sorry, could not allocate enough memory\r\n");
X#endif
X
X        Pterm(4);
X    }
X
X#else /* MWC || MANX */
X#ifdef DYNABUF
X    if((bufr = dalloc()) == (unsigned char *)NULL)
X    {
X#ifdef REMOTE
X        Bauxws("Sorry, could not allocate enough memory\r\n");
X#else
X        Bconws("Sorry, could not allocate enough memory\r\n");
X#endif
X        Pterm(5);
X    }
X#endif /* DYNABUF */
X#endif /* MWC || MANX */
X
X#ifndef REMOTE
X#if (MWC || MANX)
X    ms_ptr = (long *) ((0xffffff00L & ((long)(m_screen))) + 0x00000100L);
X#else
X    ms_ptr = (long *) ((0xffffff00L & ((long)(&m_screen[0]))) + 0x00000100L);
X#endif /* MWC */
X
X    EscSeq('e');       /* Turn on the cursor */
X    EscSeq('v');       /* wrap at end of line */
X    EscSeq('E');       /* clear screen */
X#endif /* REMOTE */
X
X    SetIoBuf();
X
X    speed = getbaud();
X/*  Txoff(); */
X    Rsconf((int) speed, flowctl, ucr, rsr, tsr, scr);  /* init set */
```

```
X    Vsync(); Vsync();
X/*  Txon();    */
X
X    Baudrate = BAUD_RATE(speed);
X
X#ifndef REMOTE
X    aline_addr = aaddress();
X    STDERR = stderr;
X#else
X#ifndef DLIBS
X    if((STDERR = fopen("aux:", "rw")) == (FILE *)NULL)
X    {
X         Bauxws("Could not Open Aux Stream for Stderr\r\n");
X         finish();
X    }
X    setbuf(STDERR, (char *)NULL);
X#else
X    STDERR = stdaux;
X#endif /* DLIBS */
X
X#endif /* remote */
X
X    help();
X
X    i = 0;
X    while (0 == 0)     /* infinite loop */
X    {
X#ifndef REMOTE
X         while (Bconstat(rs232) != 0)
X         {
X              /* Char at Modem */
X              c = Bconin(rs232) & 0x007f;
X              Bconout(console, c);
X
X              /* Check the console once in a while */
X              /* important at High speeds */
X              if ((++i) & 32)
X              {
X                  if (Bconstat(console) != 0)
X                       {
X                       conin = Bconin(console);
X                       Bconout(rs232, (int) (conin & 0x007f));
X                  }
X                  i = 0;
X              }
X
X         }
X
X         if (Bconstat(console) != 0)
X         {
X              /* Char at Console */
X              conin = Bconin(console);
X              if ((conin & 0x00FF0000L) == 0x00610000L)  /* Undo */
```

```
X             {
X                  ResetIoBuf();
X                  finish();
X             }
X
X             if ((conin & 0x00FF0000L) == 0x00620000L)  /* Help */
X                 help();
X             else
X                 Bconout(rs232, (int) (conin & 0x007f));
X         }
X#else
X         while (Bconstat(rs232) != 0)
X         {
X              /* Char at Modem */
X              c = Bconin(rs232) & 0x007f;
X
X              if ((c & CTRL('U')) == CTRL('U'))  /* Undo */
X              {
X                   ResetIoBuf();
X                   finish();
X              }
X
X              if ((c & CTRL('Z')) == CTRL('Z'))  /* Help */
X                  help();
X              else
X                  Bconout(rs232,c);
X         }
X
X#endif /* REMOTE */
X    }
X}
X
Xfinish()
X{
X#ifdef PHONES
X    /* Save phone directory if it changed */
X    extern int dchanged;
X
X    if(dchanged)
X    {
X         if(writedir() == 1)
X             hit_key();
X    }
X#endif
X
X#if (MWC || MANX)
X#ifdef DYNABUF
X    Mfree(bufr);
X#else
X    free(bufr);
X#endif
X
X#ifndef REMOTE
```

```
X    free(m_screen);
X#endif
X
X#else
X
X#ifdef DYNABUF
X    Mfree(bufr);
X#endif
X#endif
X    exit(0);
X}
X
X/* -eof- */
SHAR_EOF
echo "File MAIN.C is complete"
chmod 0600 MAIN.C || echo "restore of MAIN.C fails"
echo "x - extracting MAKEFALC.STA (Text)"
sed 's/^X//' << 'SHAR_EOF' > MAKEFALC.STA &&
X#
X# Makefile for Alcyon C (version 4.14 required) and STANDALONE versions
X#   of RZ and SZ.
X#   for use with PD MAKE distributed by us.
X#
X#   Fix paths below for your set up
X#        c:\include\   Path to include files
X#        c:\lib\       Path to C library files
X#
X#   Edit config.h before make'ing
X#
X#   Please use a decent version of gemstart.s that gives some
X#   breathing room to malloc()
X#
X#   Jwahar Bammi
X
XSRC = common.c rz.c sz.c util.c tyme.c zm.c fileio.c
XRZOBJ = common.o rz.o util.o tyme.o zm.o fileio.o
XSZOBJ = common.o sz.o util.o tyme.o zm.o fileio.o
X
XINCLUDE = c:\include
XLIB = c:\lib
XCPFLAGS = -i $(INCLUDE)\ -DDECL -DSTANDALONE
XLINKER = c:\bin\aln.prg
X
Xrz.prg : $(RZOBJ)
X    $(LINKER) -o rz.prg -c rzlnk
X
Xsz.prg : $(SZOBJ)
X    $(LINKER) -o sz.prg -c szlnk
X
X$(RZOBJ) : common.h zmdm.h decl.h config.h
X$(SZOBJ) : common.h zmdm.h decl.h config.h
X
Xclean:
```

```
X    $(RM) *.o *.68k
SHAR_EOF
chmod 0600 MAKEFALC.STA || echo "restore of MAKEFALC.STA fails"
echo "x - extracting MAKEFILE (Text)"
sed 's/^X//' << 'SHAR_EOF' > MAKEFILE &&
X#
X# Makefile for Mark Williams C
X#   Version 2.00 (or higher) required.
X#   Please ignore warnigs about a constant being promoted to long
X#   and about some unused variables.
X#
X#   Edit config.h before make'ing.
X#
X#   Jwahar Bammi
X#
X#
X
XSRC = common.c rz.c sz.c transfer.c util.c main.c tyme.c zm.c fileio.c \
Xhi5025.s phone.c
X
XOBJ =  fileio.o zm.o util.o rz.o sz.o common.o tyme.o transfer.o main.o \
Xhi5025.o phone.o
X
XCFLAGS = -O -V -VPEEP -DMWC=1
XLDFLAGS = -s -x -V
X
Xzmdm.prg : $(OBJ)
X    cc -O -o zmdm.prg $(OBJ) $(LDFLAGS)
X
X$(OBJ) : common.h zmdm.h decl.h config.h
Xtransfer.o : expandar.c
X
Xclean:
X    rm *.o
SHAR_EOF
chmod 0600 MAKEFILE || echo "restore of MAKEFILE fails"
echo "x - extracting MAKEFILE.ALC (Text)"
sed 's/^X//' << 'SHAR_EOF' > MAKEFILE.ALC &&
X#
X# Makefile for Alcyon C (version 4.14 required)
X#   for use with PD MAKE distributed by us.
X#
X#   Fix paths below for your set up
X#        c:\include\   Path to include files
X#        c:\lib\       Path to C library files
X#
X#   Edit config.h
X#
X#   Please use a decent version of gemstart.s that gives some
X#   breathing room to malloc()
X#
X#   Jwahar Bammi
X
```

```
XSRC = common.c rz.c sz.c transfer.c util.c main.c tyme.c zm.c fileio.c dummy.c \
Xphone.c
X
XOBJ = common.o rz.o sz.o transfer.o util.o main.o tyme.o zm.o fileio.o dummy.o \
Xphone.o
X
XINCLUDE = c:\include
XLIB = c:\lib
XCPFLAGS = -i $(INCLUDE)\ -DDECL
XLINKER = c:\bin\aln.prg
X
Xzmdm.prg : $(OBJ)
X    $(LINKER) -o zmdm.prg -c lnk
X
X$(OBJ) : common.h zmdm.h decl.h config.h
Xtransfer.o : expandar.c
X
Xclean:
X    $(RM) *.o *.68k
SHAR_EOF
chmod 0600 MAKEFILE.ALC || echo "restore of MAKEFILE.ALC fails"
echo "x - extracting MAKEFILE.MAN (Text)"
sed 's/^X//' << 'SHAR_EOF' > MAKEFILE.MAN &&
X#
X# Makefile for Mark Williams C
X#   Version 2.00 (or higher) required.
X#   Please ignore warnigs about a constant being promoted to long
X#   and about some unused variables.
X#
X#   Edit config.h before make'ing.
X#
X#   Jwahar Bammi
X#
X#
X
XSRC = common.c rz.c sz.c transfer.c util.c main.c tyme.c zm.c fileio.c \
Xmanx.asm phone.c
X
XOBJ =  fileio.o zm.o util.o rz.o sz.o common.o tyme.o transfer.o main.o \
Xmanx.o phone.o
X
XCFLAGS = -DMANX=1
XLDFLAGS = -V
X
Xzmdm.prg : $(OBJ)
X    ln -o zmdm.prg $(LDFLAGS) $(OBJ) -Lc
Xmanx.o : manx.asm
X    as -V manx.asm
X
X$(OBJ) : common.h zmdm.h decl.h config.h
Xtransfer.o : expandar.c
X
Xclean:
```

```
X    rm *.o
SHAR_EOF
chmod 0600 MAKEFILE.MAN || echo "restore of MAKEFILE.MAN fails"
echo "x - extracting MAKEFILE.STA (Text)"
sed 's/^X//' << 'SHAR_EOF' > MAKEFILE.STA &&
X#
X# Makefile for Mark Williams C for standalone versions of RZ and SZ
X#   Version 2.00 required.
X#   Please ignore warnigs about a constant being promoted to long
X#   and about some unused variables.
X#
X#   Edit config.h before make'ing
X#
X#   Jwahar Bammi
X#
X#
XSRC = common.c rz.c sz.c util.c tyme.c zm.c fileio.c
XRZOBJ = common.o rz.o util.o tyme.o zm.o fileio.o
XSZOBJ = common.o sz.o util.o tyme.o zm.o fileio.o
X
XCFLAGS = -O -DSTANDALONE -DMWC=1 -V -VPEEP
XLDFLAGS = -x -s -V -VPEEP
X
Xrz.prg : $(RZOBJ)
X    cc -O -o rz.prg $(RZOBJ) $(LDFLAGS)
X
X$(RZOBJ) : common.h zmdm.h decl.h config.h
X
Xsz.prg : $(SZOBJ)
X    cc -O -o sz.prg $(SZOBJ) $(LDFLAGS)
X
X$(SZOBJ) : common.h zmdm.h decl.h config.h
X
Xclean:
X    rm *.o
SHAR_EOF
chmod 0600 MAKEFILE.STA || echo "restore of MAKEFILE.STA fails"
echo "x - extracting MAKEFMAN.STA (Text)"
sed 's/^X//' << 'SHAR_EOF' > MAKEFMAN.STA &&
X#
X# Makefile for Manx Aztec C for standalone versions of RZ and SZ
X#   Version 3.6a tested.
X#
X#   Edit config.h before make'ing
X#
X#   Jwahar Bammi
X#
X#
XSRC = common.c rz.c sz.c util.c tyme.c zm.c fileio.c
XRZOBJ = common.o rz.o util.o tyme.o zm.o fileio.o
XSZOBJ = common.o sz.o util.o tyme.o zm.o fileio.o
X
XCFLAGS = -DSTANDALONE -DMANX=1
```

```
XLDFLAGS = -V
X
Xrz.prg : $(RZOBJ)
X    ln -o rz.prg $(LDFLAGS) $(RZOBJ) -Lc
X
X$(RZOBJ) : common.h zmdm.h decl.h config.h
X
Xsz.prg : $(SZOBJ)
X    ln -o sz.prg $(LDFLAGS) $(SZOBJ) -Lc
X
X$(SZOBJ) : common.h zmdm.h decl.h config.h
X
Xclean:
X    rm *.o
SHAR_EOF
chmod 0600 MAKEFMAN.STA || echo "restore of MAKEFMAN.STA fails"
echo "x - extracting MANX.ASM (Text)"
sed 's/^X//' << 'SHAR_EOF' > MANX.ASM &&
X;
X;  Make hi rez screen bios handle 50 lines of 8x8 characters
X;
X;  Adapted to Manx C use from origional PD asm posting
X;  from atari corp.
X;
X;       Jwahar Bammi
X;            usenet: cwruecmp!bammi@decvax.UUCP
X;            csnet:  bammi@cwru.edu
X;            arpa:   bammi@cwru.edu
X;            CompuServe: 71515,155
X;
X;
X    cseg
X
X    public _hi50
X
X_hi50:                  ; switch to 50 line mode
X    link a6,#0          ; routine preamble
X
X    dc.w $A000          ; get the important pointers (line A init)
X
X    movea.l    4(a1),a1      ; a1 -> 8x8 font header
X
X    move.l  72(a1),-$0A(a0)   ; v_off_ad <- 8x8 offset table addr
X    move.l  76(a1),-$16(a0)   ; v_fnt_ad <- 8x8 font data addr
X
X    move    #8,-$2E(a0)      ; v_cel_ht <- 8    8x8 cell height
X    move    #49,-$2A(a0)     ; v_cel_my <- 49   maximum cell "Y"
X    move    #640,-$28(a0)     ; v_cel_wr <- 640  offset to cell Y+1
X
X    unlk a6         ; routine postable
X    rts             ; and return
X
X
```

```
X;
X; Make hi rez screen bios handle 25 lines of 8x16 characters
X;
X
X    public _hi25
X
X_hi25:                  ; Switch to 25 lines display
X    link a6,#0          ; routine preamble
X
X    dc.w    $A000       ; get the important pointers
X
X    movea.l    8(a1),a1      ; a1 -> 8x16 font header
X
X    move.l  72(a1),-$0A(a0)   ; v_off_ad <- 8x16 offset table addr
X    move.l  76(a1),-$16(a0)   ; v_fnt_ad <- 8x16 font data addr
X
X    move    #16,-$2E(a0)      ; v_cel_ht <- 16    8x16 cell height
X    move    #24,-$2A(a0)      ; v_cel_my <- 24    maximum cell "Y"
X    move    #1280,-$28(a0)    ;  v_cel_wr <- 1280  vertical byte offset
X
X    unlk a6          ; routine postamble
X    rts              ; bye
X
X
X; return the base address of the line A variables
X
X    public _aaddress
X
X_aaddress:
X    link a6,#0
X    dc.w $A000         ; Line A trap - 0000 is init aline
X                  ; d0 and a0 now contain the address
X                  ; so we can just return and the result
X                  ; will be valid
X    unlk a6
X    rts
X
X    end
SHAR_EOF
chmod 0600 MANX.ASM || echo "restore of MANX.ASM fails"
echo "x - extracting PHONE.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > PHONE.C &&
X/*
X * Phone dialing Module (from XMDM)
X *
X *
X *      Jwahar Bammi
X *           usenet: mandrill!bammi@{decvax,sun}.UUCP
X *           csnet:  bammi@mandrill.ces.CWRU.edu
X *           arpa:   bammi@mandrill.ces.CWRU.edu
X *           CompuServe: 71515,155
X */
X
```

```
X#include "config.h"
X
X#ifndef STANDALONE
X#ifdef PHONES
X
X#include "zmdm.h"
X
Xtypedef int WORD;
Xtypedef long LONG;
X
X
X#define sendchar(c)    Bconout(1, c);
X#define clear_screen()  Bconws("\033H\033J")
X#define inv()          EscSeq('p')
X#define normal()  EscSeq('q')
X#define mvto(r,c) EscSeq('Y');Bconout(2, r+040);Bconout(2, c+040)
X#define ceol()         EscSeq('K')
X#define show_cursor()  EscSeq('e')
X#define hide_cursor()  EscSeq('f')
X
Xtypedef struct _dir {      /* The Directory type */
X    char *name;        /* Name (11 chars max) */
X    char     *number;  /* Phone # (24 chars max) */
X    WORD baud;         /* baud Rate        */
X    struct _dir *next; /* Ptr to next entry */
X} *DIR;
X
X
X    /* External Variables */
Xextern WORD speed;         /* Current Baud Rate */
Xextern WORD Baudrate;
Xextern BAUDS vbauds[];
X
XWORD dchanged = 0;              /* Has the directory been updated */
X
X    /* Globals */
Xstatic WORD ndir = 0;           /* # of entries in phone directory   */
Xstatic DIR directory = (DIR)NULL;    /* The phone directory
          */
Xstatic DIR lastdir   = (DIR)NULL;    /* Pointer to last entry
          */
Xstatic char *dirfile = (char *)NULL; /* Name of file conatining directory */
Xstatic WORD rflag = 0;          /* Read directory as yet??
          */
X
X
X/*
X * Read the Telephone directory
X *  returns -2 on read error
X *      -1 if cancelled
X *      -3 if file not found
X *       n # of entries  otherwise
X */
```

```
XWORD readdir()
X{
X    register char *filename;
X    register WORD handle;
X    register WORD nentries;
X    register DIR last;
X    register DIR present;
X    extern char *preadl();
X    extern char *malloc();
X
X    Bconws("Enter Filename of Phone Directory or <CR> to Cancel: ");
X
X    if((filename = preadl()) == (char *)NULL)
X        /* Cancelled */
X        return -1;
X
X    if((dirfile = malloc(strlen(filename)+1)) == (char *)NULL)
X    {
X         /* Out of memory */
X         Bconws("Out of Memory\r\n");
X         return 0;
X    }
X    strcpy(dirfile,filename);
X
X
X    if((handle = Fopen(filename,0)) < 0)
X        /* File does not exist */
X        return -3;
X
X    /* Read in the file */
X    if(Fread(handle, 2L, &ndir) != 2L)
X    {
X         Bconws("Error Reading ");
X         Bconws(filename);
X         Bconws("\r\n");
X         Fclose(handle);
X         return -2;
X    }
X
X    /* Read in the directory */
X    last = (DIR)NULL;
X    directory = (DIR)NULL;
X    lastdir = (DIR)NULL;
X    rflag = 1;
X
X    for(nentries = 0; nentries < ndir; nentries++)
X    {
X         /* Allocate an entry */
X         if((present = (DIR)malloc(sizeof(struct _dir))) == (DIR)NULL)
X         {
X              /* Out of memory */
X              Bconws("Out of Memory\r\n");
X              Fclose(handle);
```

```
X            return nentries;
X        }
X
X        if((present->name = malloc(12)) == (char *)NULL)
X        {
X            /* Out of memory */
X            Bconws("Out of Memory\r\n");
X            Fclose(handle);
X            return nentries;
X        }
X
X        if((present->number = malloc(25)) == (char *)NULL)
X        {
X            /* Out of memory */
X            Bconws("Out of Memory\r\n");
X            Fclose(handle);
X            return nentries;
X        }
X
X        present->next = (DIR)NULL;
X
X        /* Read in the entry */
X        if(Fread(handle,11L,present->name) != 11L)
X        {
X            Bconws("Error Reading ");
X            Bconws(filename);
X            Bconws("\r\n");
X            Fclose(handle);
X            rflag = 0;
X            freedir(directory);
X            return -2;
X
X        }
X
X        if(Fread(handle,24L,present->number) != 24L)
X        {
X            Bconws("Error Reading ");
X            Bconws(filename);
X            Bconws("\r\n");
X            Fclose(handle);
X            rflag = 0;
X            freedir(directory);
X            return -2;
X        }
X
X        if(Fread(handle,2L,&(present->baud)) != 2L)
X        {
X            Bconws("Error Reading ");
X            Bconws(filename);
X            Bconws("\r\n");
X            Fclose(handle);
X            rflag = 0;
X            freedir(directory);
```

```
X             return -2;
X        }
X
X
X        present->name[11] = '\0';
X        present->number[24] = '\0';
X
X        /* Link it on with the directory */
X        if(last == (DIR)NULL)
X             /* first entry */
X             directory = present;
X        else
X             last->next = present;
X
X        last = present;
X    }
X    lastdir = last;
X
X    return nentries;
X}
X
X/*
X * Free space allocated to a phone directory
X *
X */
Xfreedir(dir)
Xregister DIR dir;
X{
X    register DIR next;
X    register DIR present;
X
X    for(present = dir; present != (DIR)NULL; present = next)
X    {
X        next = present->next;
X        free(present->name);
X        free(present->number);
X        free(present);
X    }
X
X    if(dirfile != (char *)NULL)
X    {
X        free(dirfile);
X        dirfile   = (char *)NULL;
X    }
X
X    directory = (DIR)NULL;
X    lastdir    = (DIR)NULL;
X    ndir   = 0;
X    rflag      = 0;
X}
X
X/*
X * Write out the phone directory
```

```
X *  -returns 0 on success 1 otherwise
X *
X */
XWORD writedir()
X{
X    register DIR dir;
X    register WORD fd;
X
X    if((rflag == 0) || (dirfile == (char *)NULL) || (dchanged == 0))
X        /* Nothing to Save */
X        return 0;
X
X    /* Create/Open file for write - overwrite if it exists */
X    if ((fd = Fcreate(dirfile,0)) < 0) /* Will fail if file is present */
X    {
X         /* Overwrite existing file */
X         if((fd = Fopen(dirfile,1)) < 0)
X         {
X              Bconws("Cannot Open ");
X              Bconws(dirfile);
X              Bconws("\r\n");
X              return 1;
X         }
X    }
X
X    if(Fwrite(fd,2L,&ndir) != 2L)
X    {
X         Bconws("Error Writing ");
X         Bconws(dirfile);
X         Bconws("\r\n");
X         Fclose(fd);
X         return 1;
X    }
X
X    for(dir = directory; dir != (DIR)NULL; dir = dir->next)
X    {
X         if(Fwrite(fd,11L,dir->name) != 11L)
X         {
X              Bconws("Error Writing ");
X              Bconws(dirfile);
X              Bconws("\r\n");
X              Fclose(fd);
X              return 1;
X         }
X
X         if(Fwrite(fd,24L,dir->number) != 24L)
X         {
X              Bconws("Error Writing ");
X              Bconws(dirfile);
X              Bconws("\r\n");
X              Fclose(fd);
X              return 1;
X         }
```

```
X
X        if(Fwrite(fd,2L,&(dir->baud)) != 2L)
X        {
X             Bconws("Error Writing ");
X             Bconws(dirfile);
X             Bconws("\r\n");
X             Fclose(fd);
X             return 1;
X        }
X    }
X
X    Fclose(fd);
X    return 0;
X}
X
X/*
X * Show the phone directory
X * return the entry # or -1 if cancelled
X *
X */
XWORD showdir()
X{
X    register WORD first, last;
X    register WORD n;
X    register char *line;
X    extern WORD atoi();
X    extern char *preadl();
X
X    first = 0;
X
X    while(1)
X    {
X        again:
X         clear_screen();
X
X         mvto(0,19);
X         Bconws("Phone Directory: ");
X         Bconws(dirfile);
X         Bconws("  ");
X         printf("%d",ndir); fflush(stdout);
X         Bconws(" Entry(s)");
X
X         last = (ndir < (first + 44)) ? ndir : first + 44;
X         putdir(first,last);
X
X         /* mvto(25,0); */
X         Bconws("\r\n");
X         inv();
X         Bconws("Enter a # or <SPACE><RETURN> for Next Page or <RETURN> to
Cancel:");
X         normal();
X         Bconout(2, ' ');
X         if((line = preadl()) == (char *)NULL)
```

```
X        {
X             return -1;
X        }
X
X        if(isdigit(*line))
X        {
X             if((n = atoi(line)) >= ndir)
X             {
X                  Bconws("Invalid Number ");
X                  hit_key();
X                  goto again;
X             }
X
X             return n;
X        }
X
X        if(last == ndir)
X            first = 0;
X        else
X            first += 44;
X    }
X}
X
X/*
X * Put up directory entries on the screen
X *
X */
Xputdir(first,last)
Xregister WORD first;
Xregister WORD last;
X{
X    register DIR dir;
X    register WORD row;
X    extern DIR nth();
X
X    /* Find the first entry */
X    dir = nth(first);
X    row = (first % 44) + 1;
X
X    hide_cursor();
X    inv();
X    for(; first < last; first++)
X    {
X         mvto(row,((first & 1)?41:0));
X         if(first < 10)
X              Bconout(2, ' ');
X         printf("%d",first); fflush(stdout);
X         Bconout(2, '|');
X         putstr(dir->name,11);
X         Bconout(2, '|');
X         putstr(dir->number,24);
X         dir = dir->next;
X         if(first & 1)
```

```
X             row++;
X    }
X
X    if(first & 1)
X    {
X         mvto(row,41);
X         Bconws("  |           |                        ");
X         row++;
X    }
X
X    for(; row < 23; row++)
X    {
X         Bconws("  |           |                        ");
X         mvto(row,41);
X         Bconws("  |           |                        ");
X    }
X
X    normal();
X    show_cursor();
X}
X
X/*
X * Put a string padding to len on screen
X */
Xputstr(s,l)
Xregister char *s;
Xregister WORD l;
X{
X    register WORD pad;
X
X    Bconws(s);
X    if((pad = l - strlen(s)) <= 0)
X         return;
X    for(; pad > 0; pad--)
X         Bconout(2, ' ');
X}
X
X
X/*
X * Return the nth entry in the phone directory
X *
X */
XDIR nth(n)
Xregister WORD n;
X{
X    register WORD i;
X    register DIR dir;
X
X    for(i = 0, dir = directory; (dir != (DIR)NULL) & (i < n);
X        i++, dir = dir->next)
X        /* Skip */;
X    return(dir);
X}
```

```
X
X/*
X * Add a entry in the phonebook
X */
Xaddir()
X{
X    register DIR dir;
X    register char *s;
X    extern char *malloc(), *preadl();
X
X    /* If a directory file already exists add, else read or create */
X    if(rflag == 0)
X    {
X         switch(readdir())
X         {
X             case -3:
X              /* Doesnt exist, but we will create it when we
X                 write out the directory */
X              rflag = 1;
X              break;
X
X             case 0:
X             case -2:
X             case -1:
X              hit_key();
X              return;
X
X             default:
X              rflag = 1;
X              break;
X         }
X    }
X
X    /* Allocate space for the new entry */
X    if((dir = (DIR)malloc(sizeof(struct _dir))) == (DIR)NULL)
X    {
X         Bconws("Out of Memory\r\n");
X         hit_key();
X         return;
X    }
X
X    if((dir->name = malloc(12)) == (char *)NULL)
X    {
X         /* Out of memory */
X         Bconws("Out of Memory\r\n");
X         hit_key();
X         return;
X    }
X
X    if((dir->number = malloc(25)) == (char *)NULL)
X    {
X         /* Out of memory */
X         Bconws("Out of Memory\r\n");
```

```
X         hit_key();
X         return;
X    }
X
X
X    /* Get the entry */
X
X    do {
X         Bconws("Name: ");
X         s = preadl();
X    } while(s == (char *)NULL);
X
X    strncpy(dir->name,s,11);
X
X    do {
X         Bconws("Number: ");
X         s = preadl();
X    } while(s == (char *)NULL);
X
X    strncpy(dir->number,s,24);
X
X    do {
X         Bconws("Baud Rate: ");
X         s = preadl();
X    } while((s == (char *)NULL) || ((dir->baud = tobaud(s)) == -1));
X
X    dir->next = (DIR)NULL;
X
X    if(directory == (DIR)NULL)
X        directory = dir;
X    else
X        lastdir->next = dir;
X
X    lastdir = dir;
X    dchanged = 1;
X    ndir++;
X}
X
X
X
X/*
X * Convert a string to a baud rate
X * return int or -1 if invalid
X */
XWORD tobaud(s)
Xregister char *s;
X{
X    register WORD i;
X
X    for(i = 0; vbauds[i].sbaud != (char *)NULL; i++)
X    {
X         if(strcmp(vbauds[i].sbaud,s) == 0)
X             return vbauds[i].ibaud;
```

```
X    }
X
X    Bconws(s);
X    Bconws(": Invalid Baud Rate\r\nValid Baud Rates are:\r\n");
X    for(i = 0; vbauds[i].sbaud != (char *)NULL; i++)
X    {
X         Bconout(2, '\t');
X         if((i != 0) && (vbauds[i].ibaud != vbauds[i-1].ibaud))
X         {
X              Bconws(vbauds[i].sbaud);
X              Bconws("\r\n");
X         }
X    }
X
X    return -1;
X}
X
X/*
X * Dial a number
X */
Xdial()
X{
X    register WORD n;
X    register DIR dir;
X    extern DIR nth();
X
X    /* Has the directory been read so far */
X    if(rflag == 0)
X    {
X         /* Go read it */
X         switch(readdir())
X         {
X             case -1:
X             case -2:
X             case -3:
X             case 0:
X              rflag = 0;
X              hit_key();
X              his_screen();
X              return;
X             default:
X              break;
X         }
X    }
X
X    if((n = showdir()) == -1)
X    {
X         /* Cancelled */
X         hit_key();
X         his_screen();
X         return;
X    }
X
```

```
X    his_screen();
X    dir = nth(n);
X    if(dir->baud != speed)
X    {
X         speed = dir->baud;
X         Baudrate = jbaud(speed);
X         Rsconf(speed, -1, -1, -1, -1, -1);
X         sendchar('\r');
X         flushinput();
X    }
X    write_modem(PREDIAL, strlen(PREDIAL));
X    write_modem(dir->number,strlen(dir->number));
X    sendchar('\r');
X}
X
X
X/*
X * Re-dial the previous number
X *
X *  Does it cheaply, by sending the modem its Re-dial
X *  sequence, instead of remembering the last number dialed etc.
X *
X */
Xredial()
X{
X    his_screen();
X    write_modem(REDIAL ,strlen(REDIAL));
X}
X
X
X/*
X * Open a phone directory
X *
X */
Xopendir()
X{
X    /* if one is open, save it if changed, then deallocate memory
X     * and then open a new directory
X     */
X    if(rflag)
X    {
X         if(dchanged)
X         {
X              if(writedir() == 1)
X              {
X                   hit_key();
X                   return;
X              }
X              dchanged = 0;
X         }
X         freedir(directory);
X    }
X
```

```
X    if(readdir() <= 0)
X    {
X         hit_key();
X         return;
X    }
X}
X
X/*
X * Delete an entry.
X */
Xdelentry()
X{
X    register DIR dir, del;
X    register WORD n;
X    extern DIR nth();
X
X    if(rflag == 0)
X    {
X         Bconws("Nothing to delete\r\n");
X         hit_key();
X         return;
X    }
X
X    if((n = showdir()) == -1)
X    {
X         /* Cancelled */
X         hit_key();
X         return;
X    }
X
X    if(ndir == 1)
X    {
X         del = directory;
X         directory = (DIR)NULL;
X         lastdir = (DIR)NULL;
X    }
X    else
X    {
X         if(n == 0)
X         {
X              del = directory;
X              dir = directory->next;
X              directory = dir;
X         }
X         else
X         {
X              dir = nth(n-1);
X              del = dir->next;
X              dir->next = del->next;
X              if(lastdir == del)
X                  lastdir = dir;
X         }
X    }
```

```
X
X    free(del->name);
X    free(del->number);
X    free(del);
X    ndir--;
X    dchanged = 1;
X
X}
X
X
X/*
X * Phone services - top level
X *
X */
Xphone()
X{
X    register LONG conin;
X
X    while(1)
X    {
X         clear_screen();
X         mvto(2,32);
X         inv();
X         Bconws("Phone  Services");
X         normal();
X
X         mvto(5,0);
X
X         if(rflag)
X         {
X              Bconws("\tThe Phone Directory \"");
X              Bconws(dirfile);
X              Bconws("\" containing ");
X              printf("%d", ndir); fflush(stdout);
X              Bconws(" Entry(s) is Currently Open.\r\n\n");
X         }
X         else
X             Bconws("\tNo Phone Directory is Currently Open.\r\n\n");
X
X
X         /* Put up menu */
X         Bconws("\r\n\t");
X         EscSeq('p');       /* reverse video */
X         Bconws("Undo");
X         EscSeq('q');       /* quit reverse video */
X         Bconws(" to exit the emulator.\r\n");
X
X         Bconws("\t");
X         EscSeq('p');       /* reverse video */
X         Bconws("d");
X         EscSeq('q');       /* quit reverse video */
X         Bconws(" to dial a number.\r\n");
X
```

```
X         Bconws("\t");
X         EscSeq('p');       /* reverse video */
X         Bconws("a");
X         EscSeq('q');       /* quit reverse video */
X         Bconws(" to add an entry to the phone directory.\r\n");
X
X         Bconws("\t");
X         EscSeq('p');       /* reverse video */
X         Bconws("D");
X         EscSeq('q');       /* quit reverse video */
X         Bconws(" to delete an entry from the phone directory.\r\n");
X
X         Bconws("\t");
X         EscSeq('p');       /* reverse video */
X         Bconws("o");
X         EscSeq('q');       /* quit reverse video */
X         Bconws(" to open another phone directory.\r\n");
X
X         Bconws("\t");
X         EscSeq('p');       /* reverse video */
X         Bconws("r");
X         EscSeq('q');       /* quit reverse video */
X         Bconws(" to re-dial last number.\r\n");
X
X         Bconws("\t");
X         EscSeq('p');       /* reverse video */
X         Bconws("Return");
X         EscSeq('q');       /* quit reverse video */
X         Bconws(" to return to the emulator.\r\n\n\n\n");
X
X         /* get response */
X         conin = Bconin(2);
X
X         if ((conin & 0x00FF0000L) == 0x610000L)
X         {
X              /* He hit <UNDO> */
X              his_screen();
X              ResetIoBuf();
X              finish();
X         }
X
X         switch((WORD)(conin & 0x7f))
X         {
X             case 'd':
X              /* Dial a number */
X              dial();
X              return;
X
X             case 'D':
X              /* Delete an entry */
X              delentry();
X              break;
X
```

```
X             case 'a':
X              /* Add an entry */
X              addir();
X              break;
X
X             case 'o':
X              /* Open another phone directory */
X              opendir();
X              break;
X
X
X             case 'r':
X              /* Re-Dial # */
X              redial();
X              return;
X
X             case '\r':
X              his_screen();
X              return;
X
X             default:
X              break;
X         }
X
X    }
X
X}
X
X/*
X * Convert a baud rate to its int
X * return int or -1 if invalid
X */
XWORD jbaud(bd)
Xregister WORD bd;
X{
X    register WORD i;
X
X    for(i = 0; vbauds[i].sbaud != (char *)NULL; i++)
X    {
X         if(vbauds[i].ibaud == bd)
X             return vbauds[i].jbaud;
X    }
X    return -1;
X}
X
Xstatic char scrth[80];
X/*
X * Read a line from Standard Input and return a
X * NULL terminated pointer to it
X */
Xchar *preadl()
X{
X    /* Use the scrth bufr for storage */
```

```
X    scrth[0] = 80;
X    Cconrs(scrth);
X    Bconws("\r\n");
X    if(scrth[1] == 0)
X    {
X         /* User Cancelled */
X         Bconws("Cancelled\r\n");
X         return((char *)NULL);
X    }
X    /* Terminate string that starts at scrth[2] */
X    scrth[scrth[1]+2] = '\0';
X    return(&scrth[2]);
X}
X
X#endif /* PHONES */
X#endif /* STANDALONE */
X
X/* -eof - */
SHAR_EOF
chmod 0600 PHONE.C || echo "restore of PHONE.C fails"
echo "x - extracting RZ.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > RZ.C &&
X/*
X *               ACKNOWLEDGEMENTS
X *
X *  ZMDM was derived from rz/sz for Unix  posted by
X *  Chuck Forsberg (...!tektronix!reed!omen!caf ). We
X *  thank him for his excellent code, and for giving
X *  us permission to use and distribute his code and
X *  documentation.
X *
X *  Atari St version by:
X *       Jwahar Bammi
X *            usenet: mandrill!bammi@{decvax,sun}.UUCP
X *            csnet:  bammi@mandrill.ces.CWRU.edu
X *            arpa:   bammi@mandrill.ces.CWRU.edu
X *            CompuServe: 71515,155
X */
X
X#include "config.h"
X#define RVERSION "rz 1.14 01-15-87"
X#define RSTVERSION "rz 1.01 03-07-87"
X#define OS   "Unix V7/BSD"
X
X/* #define RDEBUG */           /* a lot of debugging garb */
X
X/*
X *  ATARI ST series implementation notes:
X *
X *       - the following command line options were removed as they
X *         were either  not applicable to the ST environment or
X *         were not deemed reasonable (by me - ofcourse).
X *            1    Not Applicable here as we have a seperate
```

```
X *               serial port.
X *          7    In this day and age? Forget it, get another m/c.
X *          a/b  Ascii/Binary - the receive mode (if not
X *               over-ridden by the sender) is automatically
X *               selected depending on the extention given
X *               in the incoming file name. This idea was
X *               present in earlier rz/sz, i wonder why such
X *               a convenient feature was dropped (Chuck ??).
X *               This feature is relevant to ZMODEM only in rz,
X *               as the sender determines the file mode in
X *               XMODEM/YMODEM transfers.
X *             B Note that `B' has a special meaning.
X *               Specifying -B will force override to
X *               binary mode for each incoming file. Useful
X *               when doing St-to-St transfers.
X *          D    There is no /dev/null on the ST's
X *          u    not applicable to TOS. Upper and lower
X *               case file names are the same. All the
X *               applicable routines like uncap() and
X *               IsAnyLower() were zapped.
X *
X *      - The    [-][v]rzCOMMAND style of invocation was dropped
X *        as there is no good way to do pipes without the
X *        microRtx kernal. All references to Pipe and popen()
X *        were zapped.
X *      - Verbose is always set to 2 by automatically, as we know that
X *        stdout != stderr. This can be overridden
X *        by specifying -q to ensure that Verbose = 0
X *      - The idea of a PUBDIR and Restricted paths in the origonal
X *        code  was dropped totally as it is not applicable
X *        to the single owner ST environment. 1 man 1 machine.
X *      - CRCTABLE is default always, hey we have plenty of memory!.
X *      - LOGFILE renamed to 'rzlog/szlog' as we don't always have
X *        a meaningful environment to pick up TMPDIR from (like when
X *        running from the desktop).
X *         - When a subdirectory in an incoming path name is not
X *        present it is created.
X *      - The file mode transmitted is 0S00 where S is derived from
X *        the Read/Write attribute of the file on the ST
X *      - When a file mode is received, only the owner bits are
X *        are checked. If it was read only (r--) on the Unix sytem
X *        then it is given read only attribute on the ST, read-write
X *        otherwise.
X *      - Of course all the I/O was completely redone on the ST.
X *      - You will find two versions of VARARGS routines like log,
X *        one that takes int args, and the other that takes long
X *        (address) args, since sizeof(int) != sizeof(long)
X *        and sizeof(int) != sizeof(pointer) on the ST.
X *
X *  ST v1.01
X *   added support for 32 bit CRC's for Zmodem ++jrb
X *
X *  ST v1.2
```

```
X *   added -B ++jrb
X *   added all the recursive stuff
X *   added remote
X */
X
X/*% cc -DNFGVMIN -DCRCTABLE -K -O % -o rz; size rz
X *
X * rz.c By Chuck Forsberg
X *
X *  cc -O rz.c -o rz       USG (3.0) Unix
X *  cc -O -DV7  rz.c -o rz      Unix V7, BSD 2.8 - 4.3
X *
X *  ln rz rb           For either system
X *
X *  ln rz /usr/bin/rzrmail      For remote mail.  Make this the
X *                     login shell. rzrmail then calls
X *                     rmail(1) to deliver mail.
X *
X *       define CRCTABLE to use table driven CRC
X *
X *  Unix is a trademark of Western Electric Company
X *
X * A program for Unix to receive files and commands from computers running
X *  Professional-YAM, PowerCom, YAM, IMP, or programs supporting XMODEM.
X *  rz uses Unix buffered input to reduce wasted CPU time.
X *
X * Iff the program is invoked by rzCOMMAND, output is piped to
X * "COMMAND filename"
X *
X *  Some systems (Venix, Coherent, Regulus) may not support tty raw mode
X *  read(2) the same way as Unix. ONEREAD must be defined to force one
X *  character reads for these systems. Added 7-01-84 CAF
X *
X *  Alarm signal handling changed to work with 4.2 BSD 7-15-84 CAF
X *
X *  NFGVMIN Added 1-13-85 CAF for PC-AT Xenix systems where c_cc[VMIN]
X *  doesn't seem to work (even though it compiles without error!).
X *
X *  USG UNIX (3.0) ioctl conventions courtesy  Jeff Martin
X */
X
X
X#include "zmdm.h"
X#include "common.h"
X#include "zmodem.h"
X
Xstatic unsigned long SaveIntr;
X
X#ifndef Vsync              /* Atari forgot these in osbind.h */
X#define Vsync()   xbios(37)
X#endif
X
X#ifndef Supexec
```

```
X        /* Some versions of osbind don't define Supexec */
X#define Supexec(X) xbios(38,X)
X#endif
X
X#if (MWC || MANX)
Xextern FILE  *fopen();
X#else
Xextern FILE  *fopen(), *fopenb();
X#endif
X
X#ifndef STANDALONE
X#define RETURN return
X#else
Xint bibis() {} /* dummy */
X#endif
X
Xstatic long start_time;
X
X/* called by simulated signal interrupt or terminate to clean things up */
Xbibi(n)
Xint n;
X{
X
X    if (Zmodem)
X         zmputs(Attn);
X    canit(); mode(0);
X    fprintf(STDERR, "\r\nrz: caught signal %d; exiting", n);
X    if (fout != -1)
X    {
X         if (stfclose(fout) != 0)
X         {
X              fprintf(STDERR, "\r\nfile close ERROR\n");
X         }
X         fout = (-1);
X
X    }
X
X#ifdef RDEBUG
X    if (logf != (FILE *)NULL)
X         fclose(logf);
X#endif
X    aexit(128+n);
X}
X
X#ifdef STANDALONE
Xmain(argc, argv)
X#else
Xdorz(argc, argv)
X#endif /* STANDALONE */
Xint argc;
Xchar **argv;
X{
X    register char *cp;
```

```
X    register int npats;
X    char **patts;
X    int exitcode;
X
X#ifdef STANDALONE
X#ifdef MWC
X    extern char *lmalloc();
X#endif
X
X    /* Set up Dta */
X    Fsetdta(&statbuf);
X
X    /* Get screen rez */
X    rez = Getrez();
X    drv_map = Drvmap();
X
X#if (MWC || MANX)
X#ifndef DYNABUF
X#ifdef MWC
X    if((bufr = (unsigned char *)lmalloc((unsigned long)BBUFSIZ))
X                        == (unsigned char *)NULL)
X#else
X    if((bufr = (unsigned char *)Malloc((unsigned long)BBUFSIZ))
X                        == (unsigned char *)NULL)
X#endif
X#else
X    if((bufr = dalloc()) == (unsigned char *)NULL)
X#endif /* DYNABUF */
X    {
X#ifdef REMOTE
X        Bauxws("Sorry, could not allocate enough memory\r\n");
X#else
X        Bconws("Sorry, could not allocate enough memory\r\n");
X#endif
X
X        Pterm(4);
X    }
X#else /* MWC || MANX */
X#ifdef DYNABUF
X    if((bufr = dalloc()) == (unsigned char *)NULL)
X    {
X#ifdef REMOTE
X        Bauxws("Sorry, could not allocate enough memory\r\n");
X#else
X        Bconws("Sorry, could not allocate enough memory\r\n");
X#endif
X        Pterm(5);
X    }
X#endif /* DYNABUF */
X#endif /* MWC || MANX */
X
X#ifndef REMOTE
X    STDERR = stderr;
```

```
X#else
X#ifndef DLIBS
X    if((STDERR = fopen("aux:", "rw")) == (FILE *)NULL)
X    {
X         Bauxws("Could not Open Aux Stream for Stderr\r\n");
X         finish();
X    }
X    setbuf(STDERR, (char *)NULL);
X#else
X    STDERR = stdaux;
X#endif /* DLIBS */
X
X#endif /* REMOTE */
X
X    {
X         int speed;
X         speed = getbaud();
X         Baudrate = BAUD_RATE(speed);
X         SetIoBuf();
X         Rsconf(speed, 0,-1,-1,-1,-1);
X         Vsync(); Vsync();
X    }
X
X#endif /* STANDALONE */
X
X    SendType = 0;
X    Rxtimeout = 100;
X    exitcode = 0;
X
X    initz();
X
X#ifndef STANDALONE
X    chkinvok(argv[0]);     /* if called as  'rb' set flag */
X#else
X    Progname = "rz";
X#endif
X
X    npats = 0;
X    SaveIntr = Setexc(0x0102, -1L);
X    BusErr   = Setexc(2, -1L);
X    AddrErr  = Setexc(3, -1L);
X    vdebug = 0;
X
X    while (--argc)
X    {
X         cp = *++argv;
X         if (*cp == '-')
X         {
X              while( *++cp)
X              {
X                   switch(*cp)
X                   {
X                   case '+':
```

```
X                    Lzmanag = ZMAPND; break;
X               case 'B':
X                    ForceBinary=TRUE; break;
X               case 'c':
X                    Crcflg=TRUE; break;
X               case 'p':
X                    Lzmanag = ZMPROT;  break;
X               case 'q':
X                    Quiet=TRUE; Verbose=0; break;
X               case 't':
X                    if (--argc < 1) {
X                         rusage();
X                         RETURN(1);
X                    }
X                    Rxtimeout = atoi(*++argv);
X                    if (Rxtimeout<10 || Rxtimeout>1000)
X                    {
X                         rusage();
X                         RETURN(1);
X                    }
X                    break;
X               case 'v':
X                    ++Verbose; break;
X               default:
X                    rusage();
X                    RETURN(1);
X               }
X          }
X     }
X     else if ( !npats && argc>0)
X     {
X          if (argv[0][0])
X          {
X               npats=argc;
X               patts=argv;
X          }
X     }
X   }
X
X   if (npats > 1)
X   {
X        rusage();
X        RETURN(1);
X   }
X
X#ifdef RDEBUG
X   if (Verbose > 2)
X   {
X        if ((logf = fopen(RLOGFILE, "a"))== (FILE *)NULL)
X        {
X             fprintf(STDERR, "Can't open log file %s\n",RLOGFILE);
X             RETURN(0200);
X        }
```

```
X         fprintf(logf, "Progname=%s\n", Progname);
X         vdebug = 1;
X    }
X#endif
X
X    if ( !Quiet)
X    {
X         if (Verbose == 0)
X              Verbose = 2;
X    }
X
X    Setexc(0x0102, bibi);
X
X    Setexc(2, buserr);
X    Setexc(3, addrerr);
X
X    if((exitcode = setjmp(abrtjmp)))
X    {
X         /* on Contrl-C */
X         canit();
X         Setexc(2, BusErr);
X         Setexc(3, AddrErr);
X         Setexc(0x0102, SaveIntr);
X         RETURN(exitcode);
X    }
X
X    if(setjmp(busjmp))
X    {
X         /* On a bus error - instead of 2 bombs */
X         fprintf(STDERR,"\r\nFATAL: Bus Error\n\n");
X#ifdef RDEBUG
X         if(logf != (FILE *)NULL)
X              fclose(logf);
X#endif
X         if(fout != -1)
X         {
X              if (stfclose(fout) != 0)
X              {
X                   fprintf(STDERR, "\r\nfile close ERROR\n");
X              }
X              fout = (-1);
X         }
X         canit();
X         Setexc(2, BusErr);
X         Setexc(3, AddrErr);
X         Setexc(0x0102, SaveIntr);
X
X         RETURN(2);
X    }
X
X    if(setjmp(addrjmp))
X    {
X         /* On address error - instead of 3 bombs */
```

```
X         fprintf(STDERR,"\r\nFATAL: Address Error\n\n");
X#ifdef RDEBUG
X         if(logf != (FILE *)NULL)
X              fclose(logf);
X#endif
X         if(fout != -1)
X         {
X              if (stfclose(fout) != 0)
X              {
X                   fprintf(STDERR, "\r\nfile close ERROR\n");
X              }
X              fout = (-1);
X         }
X         canit();
X         Setexc(2, BusErr);
X         Setexc(3, AddrErr);
X         Setexc(0x0102, SaveIntr);
X
X         RETURN(3);
X    }
X
X    mode(1);
X
X    if (wcreceive(npats, patts)==ERROR)
X    {
X         exitcode=0200;
X         canit();
X    }
X
X    mode(0);
X    if (exitcode && !Zmodem)    /* bellow again with all thy might. */
X         canit();
X
X#ifdef RDEBUG
X    if(logf != (FILE *)NULL)
X         fclose(logf);
X#endif
X
X    if(fout != -1)
X    {
X         if (stfclose(fout) != 0)
X         {
X              fprintf(STDERR, "\r\nfile close ERROR\n");
X         }
X         fout = (-1);
X    }
X    Setexc(2, BusErr);
X    Setexc(3, AddrErr);
X    Setexc(0x0102, SaveIntr);
X
X    RETURN(exitcode);
X}
X
```

```
X#ifdef STANDALONE
XRETURN(n)
Xint n;
X{
X    ResetIoBuf();
X#if (MWC || MANX)
X#ifndef DYNABUF
X    free(bufr);
X#else
X    Mfree(bufr);
X#endif
X#else
X#ifdef DYNABUF
X    Mfree(bufr);
X#endif
X#endif
X
X    exit(n);
X}
X#endif /* STANDALONE */
X
Xrusage()
X{
X    fprintf(STDERR,
X         "%s for %s by ST Enthusiasts at Case Western Reserve University\n",
X         RSTVERSION, STOS);
X    fprintf(STDERR, "\tBased on %s for %s by Chuck Forsberg\n\n",
X         RVERSION, OS);
X
X    fprintf(STDERR,"Usage:  rz [-Bpqtv]        (ZMODEM Batch)\n");
X    fprintf(STDERR,"or rb [-qtv]     (YMODEM Batch)\n");
X    fprintf(STDERR,"or rz [-cqtv] file           (XMODEM or XMODEM-1k)\n");
X    fprintf(STDERR,"     -B Force Binary Mode transfers\n");
X    fprintf(STDERR,"     -v Verbose more v's give more info\n");
X    fprintf(STDERR,"          -q Quiet suppresses verbosity\n");
X    fprintf(STDERR,"     -t TIM Change timeout to TIM tenths of seconds\n");
X    fprintf(STDERR,"     -c Use 16 bit CRC    (XMODEM)\n");
X    fprintf(STDERR,"     -p Protect existing dest. file by skipping\n");
SHAR_EOF
echo "End of part 3"
echo "File RZ.C is continued in part 4"
echo "4" > s2_seq_.tmp
exit 0


From -8969162004148542006
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,2c07dcc60d35a227,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-29 12:19:46 PST
Path: utzoo!utgpu!water!watmath!clyde!att!osu-cis!tut.cis.ohio-state.edu!mailrus!
ames!umd5!purdue!decwrl!ucbvax!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
```

Newsgroups: comp.sources.atari.st
Subject: v01i044: zmdm -- terminal program with file transfer part04/08
Keywords: mwc, alcyon, shar
Message-ID: <3506@saturn.ucsc.edu>
Date: 29 May 88 19:19:46 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 2013
Approved: koreth@ssyx.ucsc.edu

Submitted-by: bammi@mandrill.ces.cwru.edu (Jwahar R. Bammi)
Posting-number: Volume 1, Issue 44
Archive-name: zmdm/part04

```
#!/bin/sh
# this is part 4 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file RZ.C continued
#
CurArch=4
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file RZ.C"
sed 's/^X//' << 'SHAR_EOF' >> RZ.C
X    fprintf(STDERR,"       transfer if the dest. file exists (ZMODEM ONLY)\n\n");
X
X
X    if(fout != -1)
X    {
X         if (stfclose(fout) != 0)
X         {
X              fprintf(STDERR, "\r\nfile close ERROR\n");
X         }
X         fout = (-1);
X    }
X
X#ifdef RDEBUG
X    if(logf != (FILE *)NULL)
X         fclose(logf);
X#endif
X
X    return(1);
X}
X
X
X/*
X * Let's receive something already.
```

```
X */
Xwcreceive(argc, argp)
Xint argc;
Xchar **argp;
X{
X    register int c;
X
X    if (Batch || argc==0)
X    {
X         Crcflg=(Wcsmask==0377);
X         if ( !Quiet)
X#ifndef REMOTE
X              fprintf(STDERR, "\n%s: ready (CTRL-C to cancel)\n\n",
X                   Progname);
X#else
X              fprintf(STDERR, "\n%s: ready\n\n",
X                   Progname);
X#endif
X         if (c=tryz())
X         {
X              if (c == ZCOMPL)
X                   return OK;
X              if (c == ERROR)
X                   goto fubar;
X              c = rzfiles();
X              if (c)
X                   goto fubar;
X         }
X         else
X         {
X              for (;;)
X              {
X                   if (wcrxpn(secbuf)== ERROR)
X                        goto fubar;
X                   if (secbuf[0]==0)
X                        return OK;
X                   if (procheader(secbuf) == ERROR)
X                        goto fubar;
X                   if (wcrx()==ERROR)
X                        goto fubar;
X              }
X         }
X    }
X    else
X    {
X         Bytesleft = DEFBYTL; Filemode = 0; Modtime = 0L;
X
X         strcpy(Pathname, *argp);
X#ifndef REMOTE
X         fprintf(STDERR, "\n%s: ready to receive %s (CTRL-C to Cancel)\n\n",
X              Progname, Pathname);
X#else
X         fprintf(STDERR, "\n%s: ready to receive %s\n\n",
```

```
X             Progname, Pathname);
X#endif
X
X#ifdef RDEBUG
X        if(logf != (FILE *)NULL)
X             fprintf(logf, "\nrz: ready to receive %s ", Pathname);
X#endif
X
X        if((fout = stfopen(Pathname,"w")) <= 0)
X             return ERROR;
X        if (wcrx()==ERROR)
X             goto fubar;
X    }
X    return OK;
X
Xfubar:
X    canit();
X
X    if (fout != -1)
X    {
X         if (stfclose(fout) != 0)
X         {
X              fprintf(STDERR, "\r\nfile close ERROR\n");
X         }
X         fout = (-1);
X    }
X
X    return ERROR;
X}
X
X
X/*
X * Fetch a pathname from the other end as a C ctyle ASCIZ string.
X * Length is indeterminate as long as less than Blklen
X * A null string represents no more files (YMODEM)
X */
Xwcrxpn(rpn)
Xchar *rpn;   /* receive a pathname */
X{
X    register int c;
X
X    PURGELINE;
X
Xet_tu:
X    Firstsec=TRUE;  Eofseen=FALSE;
X    sendline(Crcflg?WANTCRC:NAK);
X    Lleft=0; /* Do read next time ... */
X    while ((c = wcgetsec(rpn, 100)) != 0)
X    {
X         llog( "Pathname fetch returned %d\n", c);
X         if (c == WCEOT)
X         {
X              sendline(ACK);
```

```
X             Lleft=0;  /* Do read next time ... */
X             readline(1);
X             goto et_tu;
X        }
X        return ERROR;
X    }
X    sendline(ACK);
X    return OK;
X}
X
X/*
X * Adapted from CMODEM13.C, written by
X * Jack M. Wierda and Roderick W. Hart
X */
X
Xwcrx()
X{
X    register int sectnum, sectcurr;
X    register char sendchar;
X    int cblklen;           /* bytes to dump this block */
X
X    Firstsec=TRUE;sectnum=0; Eofseen=FALSE;
X    sendchar=Crcflg?WANTCRC:NAK;
X
X    for (;;)
X    {
X         sendline(sendchar);    /* send it now, we're ready! */
X         Lleft=0;  /* Do read next time ... */
X         sectcurr=wcgetsec(secbuf, (sectnum&0177)?50:130);
X         report(sectcurr);
X         if (sectcurr==(sectnum+1 &Wcsmask))
X         {
X              sectnum++;
X              cblklen = Bytesleft>Blklen ? Blklen:Bytesleft;
X              if (putsec(secbuf, cblklen)==ERROR)
X                   return ERROR;
X              if ((Bytesleft-=cblklen) < 0)
X                   Bytesleft = 0;
X              sendchar=ACK;
X         }
X         else if (sectcurr==(sectnum&Wcsmask))
X         {
X              log2( "Received dup Sector\n");
X              sendchar=ACK;
X         }
X         else if (sectcurr==WCEOT)
X         {
X              if (closeit(0L))
X                   return ERROR;
X              sendline(ACK);
X              Lleft=0;  /* Do read next time ... */
X              return OK;
X         }
```

```
X         else if (sectcurr==ERROR)
X              return ERROR;
X         else
X         {
X              log2( "Sync Error\n");
X              return ERROR;
X         }
X    }
X}
X
X
X/*
X * Wcgetsec fetches a Ward Christensen type sector.
X * Returns sector number encountered or ERROR if valid sector not received,
X * or CAN CAN received
X * or WCEOT if eot sector
X * time is timeout for first char, set to 4 seconds thereafter
X ***************** NO ACK IS SENT IF SECTOR IS RECEIVED OK **************
X *    (Caller must do that when he is good and ready to get next sector)
X */
Xwcgetsec(rxbuf, maxtime)
Xchar *rxbuf;
Xint maxtime;
X{
X    register int checksum, wcj, firstch;
X    register unsigned int oldcrc;
X    register char *p;
X    int sectcurr;
X
X    for (Lastrx=errors=0; errors<RETRYMAX; errors++)
X    {
X
X         if ((firstch=readline(maxtime))==STX)
X         {
X              Blklen=KSIZE; goto get2;
X         }
X         if (firstch==SOH)
X         {
X              Blklen=SECSIZ;
Xget2:
X              sectcurr=readline(1);
X              if ((sectcurr+(oldcrc=readline(1)))==Wcsmask)
X              {
X                   oldcrc=checksum=0;
X                   for (p=rxbuf,wcj=Blklen; --wcj>=0; )
X                   {
X                        if ((firstch=readline(1)) < 0)
X                             goto bilge;
X                        oldcrc=updcrc(firstch, oldcrc);
X                        checksum += (*p++ = firstch);
X                   }
X                   if ((firstch=readline(1)) < 0)
X                        goto bilge;
```

```
X                  if (Crcflg)
X                  {
X                       oldcrc=updcrc(firstch, oldcrc);
X                       if ((firstch=readline(1)) < 0)
X                            goto bilge;
X                       oldcrc=updcrc(firstch, oldcrc);
X                       if (oldcrc & 0xFFFF)
X                            llog("CRC=0%o\n", oldcrc);
X                       else
X                       {
X                            Firstsec=FALSE;
X                            return sectcurr;
X                       }
X                  }
X                  else if (((checksum-firstch)&Wcsmask)==0)
X                  {
X                       Firstsec=FALSE;
X                       return sectcurr;
X                  }
X                  else
X                       log2( "Checksum Error\n");
X             }
X             else
X                  log2("Sector number garbled 0%o 0%o\n",
X                   sectcurr, oldcrc);
X        }
X        /* make sure eot really is eot and not just mixmash */
X
X        else if (firstch==EOT && Lleft==0)
X             return WCEOT;
X
X        else if (firstch==CAN)
X        {
X             if (Lastrx==CAN)
X             {
X                  log2( "Sender CANcelled\n");
X                  return ERROR;
X             }
X             else
X             {
X                  Lastrx=CAN;
X                  continue;
X             }
X        }
X        else if (firstch==TIMEOUT)
X        {
X             if (Firstsec)
X                  goto humbug;
Xbilge:
X             log2( "Timeout\n");
X        }
X        else
X             llog( "Got 0%o sector header\n", firstch);
```

```
X
Xhumbug:
X        Lastrx=0;
X        while(readline(1)!=TIMEOUT)
X             ;
X        if (Firstsec)
X        {
X             sendline(Crcflg?WANTCRC:NAK);
X             Lleft=0; /* Do read next time ... */
X        }
X        else
X        {
X             maxtime=40; sendline(NAK);
X             Lleft=0; /* Do read next time ... */
X        }
X    }
X    /* try to stop the bubble machine. */
X    canit();
X    return ERROR;
X}
X
X
X
X
X/*
X * Process incoming file information header
X */
Xprocheader(name)
Xchar *name;
X{
X    register char  *p;
X    register int dot;
X    char openmode[4];
X    extern int strlen();
X
X    /* convert to ST style path names */
X    for( p = name; *p != '\0'; p++)
X    {
X        if(*p == '/')
X             *p = '\\';
X    }
X
X    /* pick out the last extention in the filename in each part of path */
X    while(p != name)
X    {
X        dot = 0; p-- ;
X        while((p != name) && (*p != '\\'))
X        {
X             if(*p == '.')
X             {
X                  if(dot == 0)
X                  {
X                       dot = 1;
```

```
X                  }
X                  else
X                  {
X                     /* replace all but the last dot with '_' */
X                       *p = '_';
X                  }
X             }
X             p--;
X        }
X    }
X
X    /* set default parameters and overrides */
X    strcpy(openmode,"w");
X
X    Thisbinary = isbinary(name);
X
X    if (Lzmanag)
X         zmanag = Lzmanag;
X
X    /*
X     *  Process ZMODEM remote file management requests
X     */
X    if ( zconv == ZCNL)     /* Remote ASCII override */
X         Thisbinary = 0;
X    if (zconv == ZCBIN)     /* Remote Binary override */
X         ++Thisbinary;
X    else if (zmanag == ZMAPND)
X         strcpy(openmode, "a");
X
X    if (ForceBinary == TRUE )   /* local binary force override */
X         ++Thisbinary;
X
X    /* ZMPROT check for existing file */
X    if (zmanag == ZMPROT && existf(name, "r"))
X    {
X         return ERROR;
X    }
X
X/* ATARI ST NOTE:
X *  We will not accept rooted paths ie. paths that begin in '\' or '.\'
X *  If the incoming filename is rooted, we skip the beginning
X *  '\'   '.\'  or  '..\'
X */
X
X    if( (name[0] == '\\') || (name[0] == '.')  )
X    {
X         /* skip over the leading stuff */
X         if(name[0] == '\\')
X              name = &name[1];
X         else
X         {
X              if(name[1] == '.')
X                   name = &name[3];  /* Skip the "..\" */
```

```
X             else
X                  name = &name[2];  /* Skip the ".\"  */
X         }
X    }
X
X    /* ST addition, create any dierctories in the path that don't exist */
X    if( pathensure(name) == ERROR)
X         return ERROR;
X
X    Bytesleft = DEFBYTL; Filemode = 0; Modtime = 0L;
X
X    p = name + 1 + strlen(name);
X    if (*p)
X    {    /* file coming from Unix or DOS system */
X         sscanf(p, "%ld%lo%o", &Bytesleft, &Modtime, &Filemode);
X/* NA to atari
X             if (Filemode & UNIXFILE)
X                  ++Thisbinary;
X*/
X#ifndef REMOTE
X         if (Verbose)
X         {
X             fprintf(STDERR,
X             "\nIncoming:\n\tName:\t%s\n\tBytes:\t%ld\n\
X\tModTime: %ld\n\tMode:\t%o\n\tBufSize: %ld\n\n",
X               name, Bytesleft, Modtime, Filemode, (long)BBUFSIZ);
X
X#ifdef RDEBUG
X             if(logf != (FILE *)NULL)
X                  fprintf(logf,  "Incoming: %s %ld %lo %o\n",
X                    name, Bytesleft, Modtime, Filemode);
X#endif
X
X         }
X#endif /* REMOTE */
X
X    }
X    else
X    {         /* File coming from CP/M system */
X         for (p=name; *p; ++p)       /* change / to _ */
X             if ( *p == '/')
X                  *p = '_';
X
X         if ( *--p == '.')      /* zap trailing period */
X             *p = 0;
X    }
X
X    strcpy(Pathname, name);
X#ifndef REMOTE
X    if (Verbose)
X    {
X         fprintf(STDERR,  "Receiving %s %s [mode %s]\n\n",
X           name, Thisbinary?"BIN":"ASCII", openmode);
```

```
X
X#ifdef RDEBUG
X        if(logf != (FILE *)NULL)
X             fprintf(logf,  "Receiving %s %s %s\n",
X               name, Thisbinary?"BIN":"ASCII", openmode);
X#endif
X
X    }
X#endif /* REMOTE */
X
X    if ((fout=stfopen(name, openmode)) <= 0)
X         return ERROR;
X
X    return OK;
X}
X
X/*
X * Putsec writes the n characters of buf to receive file fout.
X *  If not in binary mode,  all characters
X *  starting with CPMEOF are discarded.
X */
Xputsec(buf, n)
Xunsigned char *buf;
Xregister int n;
X{
X    register unsigned char *p;
X
X    if (Thisbinary)
X    {
X         for (p=buf; --n>=0; p++ )
X         {
X              if(stputc( *p, fout) < 0)
X              {
X                   fprintf(STDERR, "\r\nError while Writing file\n");
X                   return ERROR;
X              }
X         }
X    }
X    else
X    {
X         if (Eofseen)
X              return OK;
X
X         for (p=buf; --n>=0; p++ )
X         {
X              if (*p == CPMEOF)
X              {
X                   Eofseen=TRUE;
X                   return OK;
X              }
X              if(*p == '\n')
X              {
X                   if(stputc('\r' ,fout) < 0)
```

```
X                   {
X                        fprintf(STDERR, "\r\nError while Writing file\n");
X                        return ERROR;
X                   }
X              }
X              if(stputc(*p ,fout) < 0)
X              {
X                   fprintf(STDERR,"\r\nError while Writing file\n");
X                   return ERROR;
X              }
X         }
X    }
X
X    return OK;
X}
X
X/*
X * Log an error only if high verbose
X */
X/*VARARGS1*/
Xllog(s,p,u)
Xchar *s;
Xint p, u;
X{
X    if (Verbose < 3)
X         return;
X#ifdef RDEBUG
X    fprintf(logf, "error %d: ", errors);
X    fprintf(logf, s, p, u);
X#endif
X
X    fprintf(STDERR, "\nerror %d: ", errors);
X    fprintf(STDERR, s, p, u);
X}
X
X
X
X#ifndef STANDALONE
X/*
X * If called as rb use YMODEM protocol
X */
Xchkinvok(s)
Xchar *s;
X{
X    Progname = s;
X    if (s[0]=='r' && s[1]=='b')
X         Nozmodem = TRUE;
X}
X#endif
X
X/*
X * Initialize for Zmodem receive attempt, try to activate Zmodem sender
X *  Handles ZSINIT frame
```

```
X *  Return ZFILE if Zmodem filename received, -1 on error,
X *   ZCOMPL if transaction finished,  else 0
X */
Xtryz()
X{
X    register int n;
X    register int cmdzack1flg;
X
X    if (Nozmodem)      /* Check for "rb" program name */
X         return 0;
X
X
X    for (n=Zmodem?10:5; --n>=0; )
X    {
X         /* Set buffer length (0) and capability flags */
X         stohdr(0L);
X         Txhdr[ZF0] = CANFC32|CANFDX|CANOVIO|CANBRK; /* of course we can break */
X
X         zshhdr(tryzhdrtype, Txhdr);
X/*      zshhdr(Badclose?ZFERR:ZRINIT, Txhdr); */
Xagain:
X         switch (zgethdr(Rxhdr, 0))
X         {
X         case ZRQINIT:
X              continue;
X         case ZEOF:
X              continue;
X         case TIMEOUT:
X              continue;
X         case ZFILE:
X              zconv = Rxhdr[ZF0];
X              zmanag = Rxhdr[ZF1];
X              ztrans = Rxhdr[ZF2];
X              tryzhdrtype = ZRINIT;
X              Badclose = FALSE;
X              if (zrdata(secbuf, KSIZE) == GOTCRCW)
X                   return ZFILE;
X              zshhdr(ZNAK, Txhdr);
X              goto again;
X         case ZSINIT:
X              if (zrdata(Attn, ZATTNLEN) == GOTCRCW)
X              {
X                   zshhdr(ZACK, Txhdr);
X                   goto again;
X              }
X              zshhdr(ZNAK, Txhdr);
X              goto again;
X         case ZFREECNT:
X              stohdr(~0L);
X              zshhdr(ZACK, Txhdr);
X              goto again;
X         case ZCOMMAND:
X              cmdzack1flg = Rxhdr[ZF0];
```

```
X             if (zrdata(secbuf, KSIZE) == GOTCRCW)
X             {
X                  if (cmdzack1flg & ZCACK1)
X                       stohdr(0L);
X                  else
X                       stohdr((long)sys2(secbuf));
X                  PURGELINE;    /* dump impatient questions */
X                  do {
X                       zshhdr(ZCOMPL, Txhdr);
X                  }
X                  while (++errors<10 && zgethdr(Rxhdr,1) != ZFIN);
X                  ackbibi();
X                  if (cmdzack1flg & ZCACK1)
X                       exec2(secbuf);
X                  return ZCOMPL;
X             }
X             zshhdr(ZNAK, Txhdr); goto again;
X        case ZCOMPL:
X             goto again;
X        default:
X             continue;
X        case ZFIN:
X             ackbibi(); return ZCOMPL;
X        case ZCAN:
X             return ERROR;
X        }
X   }
X   return 0;
X}
X
X/*
X * Receive 1 or more files with ZMODEM protocol
X */
Xrzfiles()
X{
X   register int c;
X
X   for (;;) {
X        switch (c = rzfile()) {
X        case ZEOF:
X        case ZSKIP:
X             switch (tryz()) {
X             case ZCOMPL:
X                  return OK;
X             default:
X                  return ERROR;
X             case ZFILE:
X                  break;
X             }
X             continue;
X        default:
X             return c;
X        case ERROR:
```

```
X             return ERROR;
X         }
X    }
X}
X
X/*
X * Receive a file with ZMODEM protocol
X *  Assumes file name frame is in secbuf
X */
Xrzfile()
X{
X    register int c, n;
X    long rxbytes;
X    extern void rd_time();
X
X    Eofseen=FALSE;
X    if (procheader(secbuf) == ERROR) {
X         return (tryzhdrtype = ZSKIP);
X/*       zshhdr(ZSKIP, Txhdr);
X         return ZSKIP; */
X    }
X
X    n = 10; rxbytes = 0L;
X
X    Supexec(rd_time);
X    start_time = pr_time;
X
X    for (;;)
X    {
X         stohdr(rxbytes);
X         zshhdr(ZRPOS, Txhdr);
Xnxthdr:
X         switch (c = zgethdr(Rxhdr, 0)) {
X         default:
X              vfile("rzfile: zgethdr returned %d", c);
X              return ERROR;
X         case ZNAK:
X         case TIMEOUT:
X              if ( --n < 0)
X              {
X                   vfile("rzfile: zgethdr returned %d", c);
X                   return ERROR;
X              }
X         case ZFILE:
X              zrdata(secbuf, KSIZE);
X              continue;
X         case ZEOF:
X              /* ++jrb */
X              if (rclhdr(Rxhdr) != rxbytes)
X              {
X                   /*
X                        * Ignore eof if it's at wrong place - force
X                        *  a timeout because the eof might have gone
```

```
X                       *  out before we sent our zrpos.
X                       */
X
X                      errors = 0;  goto nxthdr;
X             }
X/* --jrb
X             if (rclhdr(Rxhdr) != rxbytes)
X             {
X                  continue;
X             }
X-- */
X             if (closeit(rxbytes))
X             {
X                  tryzhdrtype = ZFERR;
X                  Badclose = TRUE;
X                  vfile("rzfile: closeit returned <> 0");
X                  return ERROR;
X             }
X             vfile("rzfile: normal EOF");
X             return c;
X        case ERROR:   /* Too much garbage in header search error */
X             if ( --n < 0)
X             {
X                  vfile("rzfile: zgethdr returned %d", c);
X                  return ERROR;
X             }
X             zmputs(Attn);
X             continue;
X        case ZDATA:
X             n = 10;
X             if (rclhdr(Rxhdr) != rxbytes)
X             {
X                  zmputs(Attn);
X                  continue;
X             }
Xmoredata:
X             switch (c = zrdata(secbuf, KSIZE))
X             {
X             case ZCAN:
X                  vfile("rzfile: zgethdr returned %d", c);
X                  return ERROR;
X             case ERROR:   /* CRC error */
X                  if ( --n < 0)
X                  {
X                       vfile("rzfile: zgethdr returned %d", c);
X                       return ERROR;
X                  }
X                  zmputs(Attn);
X                  continue;
X             case TIMEOUT:
X                  if ( --n < 0)
X                  {
X                       vfile("rzfile: zgethdr returned %d", c);
```

```
X                    return ERROR;
X               }
X               continue;
X          case GOTCRCW:
X               if(putsec(secbuf, Rxcount) == ERROR)
X                    return ERROR;
X               rxbytes += Rxcount;
X               lreport(rxbytes);
X               stohdr(rxbytes);
X               zshhdr(ZACK, Txhdr);
X               goto nxthdr;
X          case GOTCRCQ:
X               if(putsec(secbuf, Rxcount) == ERROR)
X                    return ERROR;
X               rxbytes += Rxcount;
X               lreport(rxbytes);
X               stohdr(rxbytes);
X               zshhdr(ZACK, Txhdr);
X               goto moredata;
X          case GOTCRCG:
X               if(putsec(secbuf, Rxcount) == ERROR)
X                    return ERROR;
X               rxbytes += Rxcount;
X               lreport(rxbytes);
X               goto moredata;
X          case GOTCRCE:
X               if(putsec(secbuf, Rxcount) == ERROR)
X                    return ERROR;
X               rxbytes += Rxcount;
X               lreport(rxbytes);
X               goto nxthdr;
X          }
X     }
X}
X
X/*
X * Send a string to the modem, processing for \336 (sleep 1 sec)
X *   and \335 (break signal)
X */
Xzmputs(s)
Xchar *s;
X{
X    register int c;
X
X    while (*s) {
X         switch (c = *s++) {
X         case '\336':
X              stsleep(1); continue;
X         case '\335':
X              sendbrk(); continue;
X         default:
X              sendline(c);
```

```
X         }
X    }
X}
X
X
X
X/*
X * Close the receive dataset, return OK or ERROR
X */
Xcloseit(rxbytes)
Xlong rxbytes;
X{
X    unsigned int timep[2];
X    long end_time;
X    extern void rd_time();
X
X    if (stfclose(fout) != 0) {
X         fprintf(STDERR, "\r\nfile close ERROR\n");
X         return ERROR;
X    }
X    fout = (-1);
X
X    Supexec(rd_time);
X    end_time = pr_time;
X
X    if (Modtime) {
X         unix2st(Modtime, &timep[0], &timep[1]);
X         touch(Pathname, timep);
X    }
X
X    /* if it is read only by owner on remote, then it is set
X     * to read only on the ST, all other file modes are
X     * irrelevant.
X     */
X    if (Filemode)
X    {
X         unsigned int fmode;
X
X         fmode = (unsigned int)(Filemode & 000777);
X         if( ((fmode & 0200) == 0) && ((fmode & 0400) != 0) )
X         {
X              /* it is readonly by owner on the remote, so
X               * make it read only on the ST too
X               */
X              Fattrib(Pathname, 1, 0x01);
X         }
X    }
X#ifndef REMOTE
X    if(rxbytes != 0L)
X        fprintf(STDERR,"\n\n%s Closed\nTransfer Time %ld secs.\tfor %ld bytes\
X\tApprox %ld cps\n\n", Pathname, (end_time - start_time)/200L, rxbytes,
Xrxbytes/((end_time - start_time)/200L));
X    else
```

```
X        fprintf(STDERR,"\n\n%s Closed\n\n", Pathname);
X#endif
X    lsct = 1;
X    return OK;
X}
X
X/*
X * Ack a ZFIN packet, let byegones be byegones
X */
Xackbibi()
X{
X    register int n;
X
X    vfile("ackbibi:");
X    Readnum = 1;
X    stohdr(0L);
X    for (n=4; --n>=0; )
X    {
X         zshhdr(ZFIN, Txhdr);
X         for (;;) {
X              switch (readline(100))
X              {
X              case 'O':
X                   readline(1);  /* Discard 2nd 'O' */
X                   /* ***** FALL THRU TO ***** */
X              case TIMEOUT:
X                   vfile("ackbibi complete");
X                   return;
X              default:
X                   break;
X              }
X         }
X    }
X}
X
X
X/*
X * Strip leading ! if present, do shell escape.
X */
Xsys2(s)
Xregister char *s;
X{
X    if (*s == '!')
X         ++s;
X    return stsystem(s);
X}
X/*
X * Strip leading ! if present, do exec.
X */
Xexec2(s)
Xchar *s;
X{
X/** Are you kidding
```

```
X    if (*s == '!')
X         ++s;
X    mode(0);
X    execl("/bin/sh", "sh", "-c", s);
X**/
X}
X
X/*
X * Touch a file
X */
X#ifndef MANX
X#undef Fdatime         /* There exist brain damaged versions of osbind.h */
X#define  Fdatime(a,b,c)     gemdos(0x57,a,b,c)
X#endif /* MANX has its _Gemdos stuff */
X
Xtouch(name, timep)
Xchar *name;
Xunsigned int *timep;
X{
X    register int handl;
X
X    if((handl = Fopen(name, 0)) < 0)
X    {
X#ifndef REMOTE
X         fprintf(STDERR,"*WARNING* Could not set file modification time for %s\n",
X              name);
X#endif
X         return;
X    }
X
X
X    Fdatime(timep, handl, 1);
X    Fclose(handl);
X}
X
X/* -eof- */
SHAR_EOF
echo "File RZ.C is complete"
chmod 0600 RZ.C || echo "restore of RZ.C fails"
echo "x - extracting RZLNK (Text)"
sed 's/^X//' << 'SHAR_EOF' > RZLNK &&
Xc:\lib\gemstart.o common.o rz.o util.o tyme.o zm.o fileio.o
Xc:\lib\osbind.o c:\lib\gemlib c:\lib\libf
SHAR_EOF
chmod 0600 RZLNK || echo "restore of RZLNK fails"
echo "x - extracting SZ.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > SZ.C &&
X/*
X *               ACKNOWLEDGEMENTS
X *
X *  ZMDM was derived from rz/sz for Unix  posted by
X *  Chuck Forsberg (...!tektronix!reed!omen!caf ). We
X *  thank him for his excellent code, and for giving
```

```
X *  us permission to use and distribute his code and
X *  documentation.
X *
X *  Atari St version by:
X *       Jwahar Bammi
X *            usenet: mandrill!bammi@{decvax,sun}.UUCP
X *            csnet:  bammi@mandrill.ces.CWRU.edu
X *            arpa:   bammi@mandrill.ces.CWRU.edu
X *            CompuServe: 71515,155
X */
X
X#include "config.h"
X#define SVERSION "sz 1.23 01-15-87"
X#define SSTVERSION "sz 1.01 03-07-87"
X#define OS   "Unix V7/BSD"
X
X#ifndef STANDALONE
X#define RETURN return
X#else
Xint bibi() {} /* dummy */
X#endif
X
X/* #define SDEBUG */
X
X/*
X * sz.c By Chuck Forsberg
X *
X *  cc -O sz.c -o sz       USG (SYS III/V) Unix
X *  cc -O -DV7  sz.c -o sz      Unix Version 7, 2.8 - 4.3 BSD
X *
X *       define CRCTABLE to use table driven CRC
X *
X *  ******* Some systems (Venix, Coherent, Regulus) do not *******
X *  ******* support tty raw mode read(2) identically to    *******
X *  ******* Unix. ONEREAD must be defined to force one     *******
X *  ******* character reads for these systems.            *******
X *
X * A program for Unix to send files and commands to computers running
X *  Professional-YAM, PowerCom, YAM, IMP, or programs supporting Y/XMODEM.
X *
X *  Sz uses buffered I/O to greatly reduce CPU time compared to UMODEM.
X *
X *  USG UNIX (3.0) ioctl conventions courtesy Jeff Martin
X *
X *  St v 1.01
X *       added support for 32 bit CRC's (Zmodem) ++jrb
X *
X */
X
X#include "zmdm.h"
X#include "common.h"
X#include "zmodem.h"
X
```

```
X#ifndef Vsync              /* Atari forgot these in osbind.h */
X#define Vsync()   xbios(37)
X#endif
X
X#ifndef Supexec
X        /* Some versions of osbind don't define Supexec */
X#define Supexec(X) xbios(38,X)
X#endif
X
X#define SLOGFILE "szlog"
X
X#define purgeline()    while(Bconstat(1)) Bconin(1)
X#define S_IFDIR 0x0010
X
X/*
X * Attention string to be executed by receiver to interrupt streaming data
X *  when an error is detected.  A pause (0336) may be needed before the
X *  ^C (03) or after it.
X */
X#ifdef READCHECK
Xchar Myattn[] = { 0 };
X#else
X#ifdef USG
Xchar Myattn[] = { 03, 0336, 0 };
X#else
Xchar Myattn[] = { 0 };
X#endif
X#endif
X
X#if (MWC || MANX)
XFILE *fopen();
X#else
XFILE *fopen(), *fopenb();
X#endif
Xstatic unsigned long SaveIntr;
Xstatic int Resuming, ForceBin;
Xstatic int in;
X
X/* called by signal interrupt or terminate to clean things up */
Xbibis(n)
Xint n;
X{
X    canit(); flush_modem(); mode(0);
X    fprintf(STDERR, "\r\nsz: caught signal %d; exiting\n", n);
X
X    aexit(128+n);
X}
X
X/* Called when Zmodem gets an interrupt (^X) */
X#ifdef ONINTR
Xonintr()
X{
X    siggi = 0;
```

```
X    longjmp(intrjmp, -1);
X}
X#endif
X
X#define ZKER
Xint Zctlesc; /* Encode control characters */
X
X#ifdef STANDALONE
Xint main(argc, argv)
X#else
Xint dosz(argc, argv)
X#endif
Xint argc;
Xchar **argv;
X{
X    register char *cp;
X    register int npats;
X    int agcnt; char **agcv;
X    char **patts;
X
X#ifdef STANDALONE
X#ifdef MWC
X    extern char *lmalloc();
X#endif
X
X    /* Set up Dta */
X    Fsetdta(&statbuf);
X
X    /* Get screen rez */
X    rez = Getrez();
X    drv_map = Drvmap();
X
X#if (MWC || MANX)
X#ifndef DYNABUF
X#ifdef MWC
X    if((bufr = (unsigned char *)lmalloc((unsigned long)BBUFSIZ))
X                        == (unsigned char *)NULL)
X#else
X    if((bufr = (unsigned char *)Malloc((unsigned long)BBUFSIZ))
X                        == (unsigned char *)NULL)
X#endif
X#else
X    if((bufr = dalloc()) == (unsigned char *)NULL)
X#endif /* DYNABUF */
X    {
X#ifdef REMOTE
X        Bauxws("Sorry, could not allocate enough memory\r\n");
X#else
X        Bconws("Sorry, could not allocate enough memory\r\n");
X#endif
X
X        Pterm(4);
X    }
```

```
X#else /* MWC || MANX */
X#ifdef DYNABUF
X    if((bufr = dalloc()) == (unsigned char *)NULL)
X    {
X#ifdef REMOTE
X        Bauxws("Sorry, could not allocate enough memory\r\n");
X#else
X        Bconws("Sorry, could not allocate enough memory\r\n");
X#endif
X        Pterm(5);
X    }
X#endif /* DYNABUF */
X#endif /* MWC || MANX */
X
X#ifndef REMOTE
X    STDERR = stderr;
X#else
X#ifndef DLIBS
X    if((STDERR = fopen("aux:", "rw")) == (FILE *)NULL)
X    {
X        Bauxws("Could not Open Aux Stream for Stderr\r\n");
X        finish();
X    }
X    setbuf(STDERR, (char *)NULL);
X#else
X    STDERR = stdaux;
X#endif /* DLIBS */
X
X#endif /* REMOTE */
X    {
X        int speed;
X        speed = getbaud();
X        Baudrate = BAUD_RATE(speed);
X        SetIoBuf();
X        Rsconf(speed, 0,-1,-1,-1,-1);
X        Vsync(); Vsync();
X    }
X#endif /* STANDALONE */
X
X    SendType = 1;
X    Rxtimeout = 600;
X    npats=0;
X    if (argc<2)
X    {
X        susage();
X        RETURN(1);
X    }
X
X    initz();
X#ifndef STANDALONE
X    schkinvok(argv[0]);
X#else
X    Progname = "sz";
```

```
X#endif
X
X    SaveIntr = Setexc(0x0102, -1L);
X    BusErr   = Setexc(2, -1L);
X    AddrErr  = Setexc(3, -1L);
X
X    Verbose = 0;
X    Resuming = FALSE;
X    ForceBin = FALSE;
X    in = (-1);
X    vdebug = 0;
X
X#ifdef SDEBUG
X    logf = (FILE *)NULL;
X#endif
X    while (--argc) {
X         cp = *++argv;
X         if (*cp++ == '-' && *cp) {
X              while ( *cp) {
X                   switch(*cp++) {
X                   case '+':
X                        Lzmanag = ZMAPND; break;
X#ifdef CSTOPB
X                   case '2':
X                        Twostop = TRUE; break;
X#endif
X                   case '7':
X                        Wcsmask=0177; break;
X
X/*
X    On the St we look up the ext and decide. For Xmodem
X    transfers, the file is always sent in binary mode
X    and it is the responsibility of the receiver to
X    strip CR if so desired.
X                   case 'a':
X                        Lzconv = ZCNL;
X                        Ascii = TRUE; break;
X                   case 'b':
X                        Lzconv = ZCBIN; break;
X*/
X
X/*  ST extention, force binary, useful to back up every thing
X *  in image mode, see -B option of rz too +jrb
X */
X                   case 'B':
X                        ForceBin = TRUE;
X                        Lzconv = ZCBIN;
X                        break;
X                   case 'C':
X                        if (--argc < 1) {
X                             susage();
X                             RETURN(1);
X                        }
```

```
X                    Cmdtries = atoi(*++argv);
X                    break;
X               case 'i':
X                    Cmdack1 = ZCACK1;
X                    /* **** FALL THROUGH TO **** */
X               case 'c':
X                    if (--argc != 1) {
X                         susage();
X                         RETURN(1);
X                    }
X                    Command = TRUE;
X                    Cmdstr = *++argv;
X                    break;
X               case 'd':
X                    ++Dottoslash;
X                    /* **** FALL THROUGH TO **** */
X               case 'f':
X                    Fullname=TRUE; break;
X               case 'E':
X                    Zctlesc = (-1); break;
X               case 'e':
X                    Zctlesc = 1; break;
X               case 'k':
X                    Blklen=KSIZE; break;
X               case 'L':
X                    if (--argc < 1) {
X                         susage();
X                         RETURN(1);
X                    }
X                    blkopt = atoi(*++argv);
X                    if (blkopt<32 || blkopt>1024)
X                    {
X                         susage();
X                         RETURN(1);
X                    }
X                    break;
X               case 'l':
X                    if (--argc < 1) {
X                         susage();
X                         RETURN(1);
X                    }
X                    Tframlen = atoi(*++argv);
X                    if (Tframlen<32 || Tframlen>1024)
X                    {
X                         susage();
X                         RETURN(1);
X                    }
X                    break;
X               case 'N':
X                    Lzmanag = ZMDIFF;  break;
X               case 'n':
X                    Lzmanag = ZMNEW;  break;
X               case 'o':
```

```
X                       Wantfcs32 = FALSE; break;
X                   case 'p':
X                       Lzmanag = ZMPROT;  break;
X                   case 'r':
X                       Lzconv = ZCRESUM; Resuming = TRUE; break;
X                   case 'q':
X                       Quiet=TRUE; Verbose=0; break;
X                   case 't':
X                       if (--argc < 1) {
X                           susage();
X                           RETURN(1);
X                       }
X                       Rxtimeout = atoi(*++argv);
X                       if (Rxtimeout<10 || Rxtimeout>1000)
X                       {
X                           susage();
X                           RETURN(1);
X                       }
X                       break;
X#ifdef TESTATTN
X                   case 'T':
X                       Testattn = TRUE; break;
X#endif
X                   case 'u':
X                       ++Unlinkafter; break;
X                   case 'v':
X                       ++Verbose; break;
X                   case 'X':
X                       ++Modem; break;
X                   case 'y':
X                       Lzmanag = ZMCLOB; break;
X                   default:
X                       susage();
X                       RETURN(1);
X                   }
X               }
X           }
X           else if ( !npats && argc>0) {
X               if (argv[0][0]) {
X                   npats=argc;
X                   patts=argv;
X               }
X           }
X       }
X       if (npats < 1 && !Command)
X       {
X           susage();
X           RETURN(1);
X       }
X
X#ifdef SDEBUG
X       if (Verbose > 2)
X       {
```

```
X         if ((logf = fopen(SLOGFILE, "a"))== (FILE *)NULL)
X         {
X              fprintf(STDERR, "Can't open log file %s\n",SLOGFILE);
X              RETURN(0200);
X         }
X         fprintf(logf, "Progname=%s\n", Progname);
X         vdebug = 1;
X         fflush(logf);
X    }
X#endif
X
X    if ( !Quiet)
X    {
X         if (Verbose < 2)
X              Verbose = 2;
X    }
X
X
X    Setexc(0x0102, bibis);
X    Setexc(2, buserr);
X    Setexc(3, addrerr);
X
X    if(setjmp(busjmp))
X    {
X         /* On a bus error - instead of 2 bombs */
X         fprintf(STDERR,"\r\nFATAL: Bus Error\n\n");
X#ifdef SDEBUG
X         if(logf != (FILE *)NULL)
X              fclose(logf);
X#endif
X         if(in != -1)
X         {
X              stfclose(in);
X              in = (-1);
X         }
X         canit();
X         Setexc(2, BusErr);
X         Setexc(3, AddrErr);
X         Setexc(0x0102, SaveIntr);
X         RETURN(2);
X    }
X
X    if(setjmp(addrjmp))
X    {
X         /* On address error - instead of 3 bombs */
X         fprintf(STDERR,"\r\nFATAL: Address Error\n\n");
X#ifdef SDEBUG
X         if(logf != (FILE *)NULL)
X              fclose(logf);
X#endif
X         if(in != -1)
X         {
X              stfclose(in);
```

```
X             in = (-1);
X        }
X        canit();
X        Setexc(2, BusErr);
X        Setexc(3, AddrErr);
X        Setexc(0x0102, SaveIntr);
X        RETURN(3);
X    }
X
X    if((Exitcode = setjmp(abrtjmp)))
X    {
X        fprintf(STDERR,"\nTransfer ABORT\n\n");
X#ifdef SDEBUG
X    if(logf != (FILE *)NULL)
X        fclose(logf);
X#endif
X        if(in != -1)
X        {
X             stfclose(in);
X             in = (-1);
X        }
X        Setexc(2, BusErr);
X        Setexc(3, AddrErr);
X        Setexc(0x0102, SaveIntr);
X        RETURN(Exitcode);
X    }
X
X    mode(1);
X
X
X    if ( !Modem) {
X        if (!Command && !Quiet && Verbose != 1)
X        {
X             fprintf(STDERR, "sz: %d file%s requested:\n",
X                  npats, npats>1?"s":"");
X             for ( agcnt=npats, agcv=patts; --agcnt>=0; )
X             {
X                  fprintf(STDERR, "%s ", *agcv++);
X             }
X             fprintf(STDERR, "\n\n");
X
X#ifdef SDEBUG
X             if(Verbose > 2)
X             {
X                  fprintf(logf, "sz: %d file%s requested:\n",
X                       npats, npats>1?"s":"");
X                  for ( agcnt=npats, agcv=patts; --agcnt>=0; )
X                  {
X                       fprintf(logf, "%s ", *agcv++);
X                  }
X                  fprintf(logf, "\n");
X                  fflush(logf);
X             }
```

```
X#endif
X        }
X        if (!Nozmodem) {
X             stohdr(0L);
X             if (Command)
X                  Txhdr[ZF0] = ZCOMMAND;
X             zshhdr(ZRQINIT, Txhdr);
X        }
X    }
X    flush_modem();
X
X    if (Command) {
X         if (getzrxinit()) {
X              Exitcode=0200; canit();
X         }
X         else if (zsendcmd(Cmdstr, 1+strlen(Cmdstr))) {
X              Exitcode=0200; canit();
X         }
X    } else if (wcsend(npats, patts)==ERROR) {
X         Exitcode=0200;
X         canit();
X    }
X    flush_modem();
X    mode(0);
X
X#ifdef SDEBUG
X    if(logf != (FILE *)NULL)
X         fclose(logf);
X#endif
X    if(in != -1)
X    {
X         fclose(in);
X         in = (-1);
X    }
X
X    putc('\n', STDERR);
X    Setexc(2, BusErr);
X    Setexc(3, AddrErr);
X    Setexc(0x0102, SaveIntr);
X    RETURN((errcnt != 0) | Exitcode);
X
X}
X
X#ifdef STANDALONE
XRETURN(n)
Xint n;
X{
X    ResetIoBuf();
X#if (MWC || MANX)
X#ifndef DYNABUF
X    free(bufr);
X#else
X    Mfree(bufr);
```

```
X#endif
X#else
X#ifdef DYNABUF
X    Mfree(bufr);
X#endif
X#endif
X    exit(n);
X}
X#endif /* STANDALONE */
X
Xwcsend(argc, argp)
Xchar *argp[];
X{
X    register int n;
X
X    Crcflg=FALSE;
X    Firstsec=TRUE;
X    for (n=0; n<argc; ++n) {
X         Totsecs = 0;
X         if (wcs(argp[n])==ERROR)
X              return ERROR;
X    }
X    Totsecs = 0;
X    if (Filcnt==0) {   /* bitch if we couldn't open ANY files */
X         if (1) {
X              Command = TRUE;
X              Cmdstr = "echo \"sz: Can't open any requested files\"";
X              if (getzrxinit()) {
X                   Exitcode=0200; canit();
X              }
X              else if (zsendcmd(Cmdstr, 1+strlen(Cmdstr))) {
X                   Exitcode=0200; canit();
X              }
X              Exitcode = 1; return OK;
X         }
X         canit();
X         fprintf(STDERR,"\n\nCan't open any requested files.\n\n");
X         return ERROR;
X    }
X    if (Zmodem)
X         saybibi();
X    else
X         wctxpn("");
X    return OK;
X}
X
Xwcs(oname)
Xchar *oname;
X{
X    extern struct stat statbuf;
X    char name[PATHLEN];
X
X    strcpy(name, oname);
```

```
X    ++Noeofseen;  Lastread = 0;  Lastc = (-1); Dontread = FALSE;
X
X    /* Check for directory or block special files */
X    if(Fsfirst(name,(int)(0x01 | 0x010 | 0x020)) != 0)
X    {
X         ++errcnt;
X         return OK;    /* may be others */
X    }
X
X    if (statbuf.st_mode & S_IFDIR ) {
X         return OK;
X    }
X
X    if((in = stfopen(oname,"r")) <= 0){
X         ++errcnt;
X         return OK;    /* pass over it, there may be others */
X    }
X
X    ++Filcnt;
X    switch (wctxpn(name)) {
X    case ERROR:
X         return ERROR;
X    case ZSKIP:
X         return OK;
X    }
X    if (!Zmodem && wctx()==ERROR)
X         return ERROR;
X    if (Unlinkafter)
X         unlink(oname);
X    return 0;
X}
X
X#define ISDRIVE(X) ( (((X >= 'a') && (X <= 'n'))) || ((X >= 'A') && (X <= 'N')))
X/*
X * generate and transmit pathname block consisting of
X *  pathname (null terminated),
X *  file length, mode time and file mode in octal
X *  as provided by the St's Fsfirst() call.
X *  N.B.: modifies the passed name, may extend it!
X */
Xwctxpn(name)
Xchar *name;
X{
X    register char *p, *q;
X    char name2[PATHLEN];
X    unsigned long unixtime;
X    extern struct stat statbuf;
X    extern unsigned long st2unix();  /* Convert St's date and time to unix
X                      time (seconds since Jan 1 1970 00:00:00) */
X
X    if(*name)
X         if(Fsfirst(name,(int)(0x01 | 0x020)) != 0)
X              return ERROR;
```

```
X
X    if (Modem) {
X#ifndef REMOTE
X        if (*name) {
X             fprintf(STDERR,
X             "Outgoing:\n\t Name: %s\n\t Size: %ld Bytes\n\
X\tBlocks: %ld\n\tBufSize: %ld\n\n",
X               name, statbuf.st_size, statbuf.st_size>>7, (long)BBUFSIZ);
X        }
X#endif /* REMOTE */
X        return OK;
X    }
X
X    vfile2("\r\nAwaiting pathname nak for %s\r\n", *name?name:"<END>");
X
X    if ( !Zmodem)
X        if (getnak())
X             return ERROR;
X
X    /* convert to Unix style path names */
X    /* skip any device identifier */
X    if(ISDRIVE(name[0]) && (name[1] == ':'))
X        name = &name[2];
X
X    for(p = name; *p != '\0'; p++)
X    {
X        if(*p == '\\')
X             *p = '/';
X    }
X
X    if(!Resuming)
X    {
X        if(ForceBin)
X        {
X             Lzconv = ZCBIN;
X             Ascii = FALSE;
X        }
X        else
X        {
X             if(!isbinary(name))
X             {
X                  /* We indicate to the other side */
X                  Lzconv = ZCNL;
X                  Ascii = TRUE;
X             }
X             else
X             {
X                  Lzconv = ZCBIN;
X                  Ascii = FALSE;
X             }
X        }
X    }
X
```

```
X    q = (char *) 0;
X    if (Dottoslash) {      /* change . to . */
X         for (p=name; *p; ++p) {
X              if (*p == '/')
X                   q = p;
X              else if (*p == '.')
X                   *(q=p) = '/';
X         }
X         if (q && strlen(++q) > 8) { /* If name>8 chars */
X              q += 8;            /*   make it .ext */
X              strcpy(name2, q);  /* save excess of name */
X              *q = '.';
X              strcpy(++q, name2);    /* add it back */
X         }
X    }
X
X    for (p=name, q=secbuf ; *p; )
X         if ((*q++ = *p++) == '/' && !Fullname)
X              q = secbuf;
X    *q++ = 0;
X    p=q;
X    while (q < (secbuf + KSIZE))
X         *q++ = 0;
X    if (*name)
X    {
X         unixtime = st2unix(statbuf.st_time, statbuf.st_date);
X         sprintf(p, "%lu %lo %o", statbuf.st_size,
X              unixtime,
X               ((statbuf.st_mode & 0x01)?0444:0644));
X    }
X
X    if(Verbose)
X#ifndef REMOTE
X         fprintf(STDERR,
X"Outgoing: [Hit CTRL-C to Cancel]\n\tName: %s\n\tSize: %ld Bytes\n\tBufSize:\
X %ld\n",
X              name, statbuf.st_size, (long)BBUFSIZ);
X    if(!Resuming)
X    {
X         fprintf(STDERR,"\tMode: %s\n\n", (Ascii)?"ASCII":"BINARY");
X    }
X    else
X    {
X         fprintf(STDERR,"\tMode: Resume Transfer Mode\n\n");
X    }
X#endif
X
X#ifdef SDEBUG
X    if(Verbose > 2)
X    {
X         fprintf(STDERR,"File: %s (%s)\n", name, p);
X         fprintf(logf,"File: %s (%s)\n", name, p);
X         fflush(logf);
```

```
X    }
X#endif
X
X    /* force 1k blocks if name won't fit in 128 byte block */
X    if (secbuf[125])
X         Blklen=KSIZE;
X    else {        /* A little goodie for IMP/KMD */
X         if (Zmodem)
X              Blklen = SECSIZ;
X         secbuf[127] = (statbuf.st_size + 127) >>7;
X         secbuf[126] = (statbuf.st_size + 127) >>15;
X    }
X    if (Zmodem)
X         return zsendfile(secbuf, (int)(1+strlen(p)+
X                   (int)((long)p-(long)secbuf)), statbuf.st_size);
X    if (wcputsec(secbuf, 0, SECSIZ)==ERROR)
X         return ERROR;
X    return OK;
X}
X
Xgetnak()
X{
X    register int firstch;
X
X    Lastrx = 0;
X    for (;;) {
X         switch (firstch = readock(800,1)) {
X         case ZPAD:
X              if (getzrxinit())
X                   return ERROR;
X              Ascii = 0;
X              return FALSE;
X         case TIMEOUT:
X              vfile("Timeout on pathname\n");
X              return TRUE;
X         case WANTG:
X#ifdef USG
X              mode(2); /* Set cbreak, XON/XOFF, etc. */
X#endif
X              Optiong = TRUE;
X              Blklen=KSIZE;
X         case WANTCRC:
X              Crcflg = TRUE;
X         case NAK:
X              return FALSE;
X         case CAN:
X              if ((firstch = readock(20,1)) == CAN && Lastrx == CAN)
X                   return TRUE;
X         default:
X              break;
X         }
X         Lastrx = firstch;
X    }
```

```
X}
X
X
Xwctx()
X{
X    register int sectnum, attempts, firstch;
X
X    Firstsec=TRUE;
X
X    while ((firstch=readock(Rxtimeout, 2))!=NAK && firstch != WANTCRC
X      && firstch != WANTG && firstch!=TIMEOUT && firstch!=CAN)
X        ;
X    if (firstch==CAN) {
X         fprintf(STDERR, "\r\nReceiver CANcelled\n");
X         return ERROR;
X    }
X    if (firstch==WANTCRC)
X         Crcflg=TRUE;
X    if (firstch==WANTG)
X         Crcflg=TRUE;
X    sectnum=1;
X    while (filbuf(secbuf, Blklen)) {
X         if (wcputsec(secbuf, sectnum, Blklen)==ERROR) {
X              return ERROR;
X         } else
X              sectnum++;
X    }
X
X#ifndef REMOTE
X    if (Verbose>1)
X    {
X         fprintf(STDERR, "\nClosing\n\n");
X#ifdef SDEBUG
X         if(Verbose > 2)
X         {
X              fprintf(logf, " Closing\n");
X              fflush(logf);
X         }
X#endif
X    }
X#endif /* REMOTE */
X
X    stfclose(in);
X    in = (-1);
X
X    attempts=0;
X    do {
X         vfile(" EOT ");
X         purgeline();
X         sendline(EOT);
X         flush_modem();
X         ++attempts;
X    }
```

```
X         while ((firstch=(readock(Rxtimeout, 1)) != ACK) && attempts < RETRYMAX);
X    if (attempts == RETRYMAX) {
X         fprintf(STDERR, "\r\nNo ACK on EOT\n");
X         return ERROR;
X    }
X    else
X         return OK;
X}
X
Xwcputsec(buf, sectnum, cseclen)
Xchar *buf;
Xint sectnum;
Xint cseclen; /* data length of this sector to send */
X{
X    register int checksum, wcj;
X    register char *cp;
X    unsigned int oldcrc;
X    int firstch;
X    int attempts;
X
X    firstch=0;    /* part of logic to detect CAN CAN */
X
X#ifndef REMOTE
X    if (Verbose>1)
X    {
X         fprintf(STDERR, "\rBlock %06d  %04dK ", Totsecs, (Totsecs>>3));
X#ifdef SDEBUG
X         if(Verbose > 2)
X         {
X              fprintf(logf, "\rBlock %d %dK ", Totsecs, (Totsecs>>3) );
X              fflush(logf);
X         }
X#endif
X    }
X#endif /* REMOTE */
X
X    for (attempts=0; attempts <= RETRYMAX; attempts++) {
X         Lastrx= firstch;
X         sendline(cseclen==KSIZE?STX:SOH);
X         sendline(sectnum);
X         sendline(-sectnum -1);
X         oldcrc=checksum=0;
X         for (wcj=cseclen,cp=buf; --wcj>=0; ) {
X              sendline(*cp);
X              oldcrc=updcrc((0377& *cp), oldcrc);
X              checksum += *cp++;
X         }
X         if (Crcflg) {
X              oldcrc=updcrc(0,updcrc(0,oldcrc));
X              sendline((int)oldcrc>>8);
X              sendline((int)oldcrc);
X         }
X         else
```

```
X             sendline(checksum);
X
X         if (Optiong) {
X             Firstsec = FALSE; return OK;
X         }
X         firstch = readock(Rxtimeout, (Noeofseen&&sectnum) ? 2:1);
Xgotnak:
X         switch (firstch) {
X         case CAN:
X             if(Lastrx == CAN) {
Xcancan:
X                 fprintf(STDERR, "\r\nCancelled\n");  return ERROR;
X             }
X             break;
X         case TIMEOUT:
X             vfile("Timeout on sector ACK\n"); continue;
X         case WANTCRC:
X             if (Firstsec)
X                 Crcflg = TRUE;
X         case NAK:
X             vfile("NAK on sector\n"); continue;
X         case ACK:
X             Firstsec=FALSE;
X             Totsecs += (cseclen>>7);
X             return OK;
X         case ERROR:
X             vfile("Got burst for sector ACK\n"); break;
X         default:
X             vfile("Got %02x for sector ACK\n", firstch); break;
X         }
X         for (;;) {
X             Lastrx = firstch;
X             if ((firstch = readock(Rxtimeout, 2)) == TIMEOUT)
X                 break;
X             if (firstch == NAK || firstch == WANTCRC)
X                 goto gotnak;
X             if (firstch == CAN && Lastrx == CAN)
X                 goto cancan;
X         }
X    }
X    fprintf(STDERR, "\r\nRetry Count Exceeded\n");
X    return ERROR;
X}
X
X/* fill buf with count chars padding with ^Z for CPM */
Xfilbuf(buf, count)
Xregister unsigned char *buf;
Xregister int count;
X{
X    register int m;
X
X    if((buf[0] = stgetc(in)) == EOF)
X         return 0;
```

```
X
X    for(m = 1; (m < count) && ((buf[m] = stgetc(in)) != EOF); m++)
X         /* loop */ ;
X
X    while (m < count)
X         buf[m++] = 032;
X
X    return count;
X
X}
X/* fill buf with count chars */
Xzfilbuf(buf, count)
Xregister char *buf;
X{
X    register int c, m;
X
X    m=count;
X    while ((c=stgetc(in))!=EOF) {
X         *buf++ =c;
X         if (--m == 0)
X              break;
X    }
X    return (count - m);
X}
X
X/*
X * readock(timeout, count) reads character(s) from file descriptor 0
X *  (1 <= count <= 3)
X * it attempts to read count characters. If it gets more than one,
X * it is an error unless all are CAN
X * (otherwise, only normal response is ACK, CAN, or C)
X *  Only looks for one if Optiong, which signifies cbreak, not raw input
X *
X * timeout is in tenths of seconds
X */
Xreadock(timeout, count)
Xint timeout, count;
X{
X    register int c;
X    static char byt[5];
X
X    if (Optiong)
X         count = 1;    /* Special hack for cbreak */
X
X    if (setjmp(tohere)) {
X         vfile("TIMEOUT\n");
X         return TIMEOUT;
X    }
X    c = timeout >> 3;
X    if (c<2)
X         c=2;
X#ifdef SDEBUG
X    if (Verbose>3) {
```

```
X         fprintf(STDERR, "Timeout=%d Calling alarm(%d) ", timeout, c);
X         byt[1] = 0;
X         fprintf(logf, "Timeout=%d Calling alarm(%d) ", timeout, c);
SHAR_EOF
echo "End of part 4"
echo "File SZ.C is continued in part 5"
echo "5" > s2_seq_.tmp
exit 0


From -4428457215147566732
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,2cad8c292d92c56f,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-29 12:21:04 PST
Path: utzoo!utgpu!water!watmath!clyde!att!osu-cis!tut.cis.ohio-state.edu!mailrus!
ames!umd5!purdue!decwrl!ucbvax!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i045: zmdm -- terminal program with file transfer  part05/08
Keywords: mwc, alcyon, shar
Message-ID: <3507@saturn.ucsc.edu>
Date: 29 May 88 19:21:04 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1979
Approved: koreth@ssyx.ucsc.edu

Submitted-by: bammi@mandrill.ces.cwru.edu (Jwahar R. Bammi)
Posting-number: Volume 1, Issue 45
Archive-name: zmdm/part05

#!/bin/sh
# this is part 5 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file SZ.C continued
#
CurArch=5
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file SZ.C"
sed 's/^X//' << 'SHAR_EOF' >> SZ.C
X         fflush(logf);
X
X    }
X#endif
X    stalarm(c);
```

```
X    c=read_modem(byt, count);
X
X    stalarm(0);
X#ifdef SDEBUG
X    if (Verbose>5)
X    {
X         fprintf(STDERR, "ret cnt=%d %x %x\n", c, byt[0], byt[1]);
X         fprintf(logf, "ret cnt=%d %x %x\n", c, byt[0], byt[1]);
X         fflush(logf);
X    }
X#endif
X
X    if (c<1)
X         return TIMEOUT;
X    if (c==1)
X         return (byt[0]&0377);
X    else
X         while (c)
X              if (byt[--c] != CAN)
X                   return ERROR;
X    return CAN;
X}
X
Xsreadline(n)
X{
X    return (readock(n, 1));
X}
X
X
Xsusage()
X{
X    fprintf(STDERR,"\nSend file(s) with ZMODEM/YMODEM/XMODEM Protocol\n");
X    fprintf(STDERR,"   (Y) = Option applies to YMODEM only\n");
X    fprintf(STDERR,"   (Z) = Option applies to ZMODEM only\n");
X    fprintf(STDERR,
X    "%s for %s by ST Enthusiasts at Case Western Reserve University\n",
X       SSTVERSION, STOS);
X    fprintf(STDERR,"\tBased on %s for %s by Chuck Forsberg\n\n", SVERSION, OS);
X/*  fprintf(STDERR,"Usage: sz [-12+adefknquvXy] [-] file ...\n"); */
X    fprintf(STDERR,"Usage: sz [-+defknquvXyB] file ...\n");
X/*  fprintf(STDERR,"   sz [-1eqv] -c COMMAND\n"); */
X    fprintf(STDERR,"   sz [-eqv] -c COMMAND\n");
X/*  fprintf(STDERR,"   1 Use stdout for modem input\n"); */
X#ifdef CSTOPB
X    fprintf(STDERR,"   2 Use 2 stop bits\n");
X#endif
X    fprintf(STDERR,"   + Append to existing destination file (Z)\n");
X/*  fprintf(STDERR,"   a (ASCII) change NL to CR/LF\n"); */
X    fprintf(STDERR,"   c send COMMAND (Z)\n");
X    fprintf(STDERR,"   d Change '.' to '/' in pathnames (Y/Z)\n");
X    fprintf(STDERR,"   e Escape control characters (Z)\n");
X    fprintf(STDERR,"   f send Full pathname (Y/Z)\n");
X    fprintf(STDERR,"   i send COMMAND, ack Immediately (Z)\n");
```

```
X    fprintf(STDERR,"   k Send 1024 byte packets (Y)\n");
X    fprintf(STDERR,"   L N Limit packet length to N bytes (Z)\n");
X    fprintf(STDERR,"   l N Limit frame length to N bytes (l>=L) (Z)\n");
X    fprintf(STDERR,"   n send file if Newer|longer (Z)\n");
X    fprintf(STDERR,"   N send file if different length|date (Z)\n");
X    fprintf(STDERR, "  o Use 16 bit CRC instead of 32 bit CRC (Z)\n");
X    fprintf(STDERR,"   p Protect existing destination file (Z)\n");
X    fprintf(STDERR,"   r Resume/Recover interrupted file transfer (Z)\n");
X    fprintf(STDERR,"   q Quiet (no progress reports)\n");
X    fprintf(STDERR,"   u Unlink file after transmission\n");
X    fprintf(STDERR,"   v Verbose - debugging information\n");
X    fprintf(STDERR,"   X XMODEM protocol - send no pathnames\n");
X    fprintf(STDERR,"   y Yes, overwrite existing file (Z)\n");
X    fprintf(STDERR,"   B Force Binary mode transfers (Z)\n");
X/*  fprintf(STDERR,
X"- as pathname sends standard input as sPID.sz or environment ONAME\n"); */
X    return(1);
X}
X
X/*
X * Get the receiver's init parameters
X */
Xgetzrxinit()
X{
X    register int n;
X
X    for (n=10; --n>=0; ) {
X
X         switch (zgethdr(Rxhdr, 1)) {
X         case ZCHALLENGE:   /* Echo receiver's challenge numbr */
X              stohdr(Rxpos);
X              zshhdr(ZACK, Txhdr);
X              continue;
X         case ZCOMMAND:          /* They didn't see out ZRQINIT */
X              stohdr(0L);
X              zshhdr(ZRQINIT, Txhdr);
X              continue;
X         case ZRINIT:
X              Rxflags = 0377 & Rxhdr[ZF0];
X              Txfcs32 = (Wantfcs32 && (Rxflags & CANFC32));
X
X              Rxbuflen = (0337 & Rxhdr[ZP0])+((0377 & Rxhdr[ZP1])<<8);
X              vfile("Rxbuflen=%d Tframlen=%d", Rxbuflen, Tframlen);
X              siggi = 0;
X#ifndef READCHECK
X#ifdef USG
X              mode(2); /* Set cbreak, XON/XOFF, etc. */
X#else
X              /* Use 1024 byte frames if no sample/interrupt */
X              if (Rxbuflen < 32 || Rxbuflen > 1024) {
X                   Rxbuflen = 1024;
X                   vfile("Rxbuflen=%d", Rxbuflen);
X              }
```

```
X#endif
X#endif
X             /* Override to force shorter frame length */
X             if (Rxbuflen && (Rxbuflen>Tframlen) && (Tframlen>=32))
X                  Rxbuflen = Tframlen;
X             if ( !Rxbuflen && (Tframlen>=32) && (Tframlen<=1024))
X                  Rxbuflen = Tframlen;
X             vfile("Rxbuflen=%d", Rxbuflen);
X
X             /*
X              * If input is not a regular file, force ACK's each 1024
X              *  (A smarter strategey could be used here ...)
X              */
X             if ((Rxbuflen == 0) || (Rxbuflen > 1024))
X                  Rxbuflen = 1024;
X             vfile("Rxbuflen=%d", Rxbuflen);
X
X             return (sendzsinit());
X         case ZCAN:
X         case TIMEOUT:
X             return ERROR;
X         case ZRQINIT:
X             if (Rxhdr[ZF0] == ZCOMMAND)
X                  continue;
X         default:
X             zshhdr(ZNAK, Txhdr);
X             continue;
X         }
X    }
X    return ERROR;
X}
X
X/* Send send-init information */
Xsendzsinit()
X{
X    register int c;
X    register int errors;
X
X    if (Myattn[0] == '\0')
X         return OK;
X
X    errors = 0;
X    for (;;) {
X         stohdr(0L);
X         zsbhdr(ZSINIT, Txhdr);
X         zsdata(Myattn, 1+strlen(Myattn), ZCRCW);
X         c = zgethdr(Rxhdr, 1);
X         switch (c) {
X         case ZCAN:
X             return ERROR;
X         case ZACK:
X             return OK;
X         default:
```

```
X             if (++errors > 9)
X                  return ERROR;
X             continue;
X         }
X    }
X}
X
X/* Send file name and related info */
Xzsendfile(buf, blen, szbytes)
Xchar *buf;
Xint blen;
Xlong szbytes;
X{
X    register int c, szstat;
X    long start_time, end_time;
X    extern void rd_time();
X
X    Supexec(rd_time);
X    start_time = pr_time;
X
X    for (;;) {
X         Txhdr[ZF0] = Lzconv;    /* file conversion request */
X         Txhdr[ZF1] = Lzmanag;   /* file management request */
X         Txhdr[ZF2] = Lztrans;   /* file transport request */
X         Txhdr[ZF3] = 0;
X         zsbhdr(ZFILE, Txhdr);
X         zsdata(buf, blen, ZCRCW);
Xagain:
X         c = zgethdr(Rxhdr, 1);
X         switch (c) {
X         case ZRINIT:
X              goto again;
X         case ZCAN:
X         case TIMEOUT:
X         case ZABORT:
X         case ZFIN:
X              return ERROR;
X         case ZSKIP:
X#ifndef REMOTE
X              fprintf(STDERR,"\n\n");
X#endif
X              stfclose(in); in = (-1); return c;
X         case ZRPOS:
X              if(stfseek(in, Rxpos, 0))
X              {
X                   fprintf(STDERR,"\r\nError While Seeking file\n");
X                   return ERROR;
X              }
X              Txpos = Rxpos; Lastc = (-1); Dontread = FALSE;
X              if((szstat =  zsendfdata()) == OK)
X              {
X#ifndef REMOTE
X                   Supexec(rd_time);
```

```
X                  end_time = pr_time;
X                  fprintf(STDERR,"%ld Bytes Sent\t\
XTransfer Time %ld secs.\tApprox %ld cps\n\n", szbytes, (end_time - start_time)/
200L,
Xszbytes/((end_time - start_time)/200L));
X#endif
X             }
X             return szstat;
X        case ERROR:
X        default:
X             continue;
X        }
X    }
X}
X
X/* Send the data in the file */
Xzsendfdata()
X{
X    register int c, c1, e;
X    register int newcnt;
X    register long tcount = 0;
X    static int tleft = 6;   /* Counter for test mode */
X
X    if (Baudrate > 300)
X         Blklen = 256;
X    if (Baudrate > 2400)
X         Blklen = KSIZE;
X/*  if (Rxbuflen && Blklen>Rxbuflen)  --jrb do not limit block size */
X    if (Rxbuflen)
X         Blklen = Rxbuflen;
X    if (blkopt && Blklen > blkopt)
X         Blklen = blkopt;
X    vfile("Rxbuflen=%d Blklen=%d", Rxbuflen, Blklen);
Xsomemore:
X    if (setjmp(intrjmp)) {
Xwaitack:
X        switch (c1 = getinsync()) {
X        default:
X        case ZCAN:
X             fprintf(STDERR,"\r\nReceiver Cancelled Transfer\n\n");
X             stfclose(in);
X             in = (-1);
X             return ERROR;
X        case ZSKIP:
X#ifndef REMOTE
X             fprintf(STDERR,"\r\nReceiver forced SKIP Transfer(1)\n\n");
X#endif
X             stfclose(in);
X             in = (-1);
X             return c1;
X        case ZACK:
X             tcount += c;
X             lreport(tcount);
```

```
X             /* fall thru */
X         case ZRPOS:
X             break;
X         case ZRINIT:
X             return OK;
X         }
X    }
X
X    siggi = 1;
X    newcnt = Rxbuflen;
X    stohdr(Txpos);
X    zsbhdr(ZDATA, Txhdr);
X
X    /*
X     * Special testing mode.  This should force receiver to Attn,ZRPOS
X     *  many times.  Each time the signal should be caught, causing the
X     *  file to be started over from the beginning.
X     */
X#ifdef TESTATTN
X    if (Testattn) {
X         if ( --tleft)
X             while (tcount < 20000) {
X                  wr_modem(qbf);
X                  tcount += strlen(qbf);
X#ifdef READCHECK
X                  while (Bconstat(1)) {
X                       switch (sreadline(1)) {
X                       case CAN:
X                       case ZPAD:
X                            goto waitack;
X                       }
X                  }
X#endif
X             }
X         siggi = 0; canit();
X         stsleep(3); purgeline(); mode(0);
X#ifdef SDEBUG
X         if(logf != (FILE *)NULL)
X             fclose(logf);
X#endif
X
X/*       printf("\nsz: Tcount = %ld\n", tcount); */
X         if (tleft) {
X             /* printf("\r\nERROR: Interrupts Not Caught\n"); */
X             aexit(1);
X         }
X         aexit(0);
X    }
X#endif
X
X    do {
X         if (Dontread) {
X             c = Lastc;
```

```
X         } else {
X              c = zfilbuf(secbuf, Blklen);
X              Lastread = Txpos;  Lastc = c;
X         }
X#ifdef SDEBUG
X         if (Verbose > 10)
X              vfile("Dontread=%d c=%d", Dontread, c);
X#endif
X
X         Dontread = FALSE;
X         if (c < Blklen)
X              e = ZCRCE;
X         else if (Rxbuflen && (newcnt -= c) <= 0)
X              e = ZCRCW;
X         else
X              e = ZCRCG;
X         zsdata(secbuf, c, e);
X         Txpos += c;
X         if(e == ZCRCG)
X         {
X              tcount += c;
X              lreport(tcount);
X         }
X
X         if (e == ZCRCW)
X              goto waitack;
X#ifdef READCHECK
X         /*
X          * If the reverse channel can be tested for data,
X          *  this logic may be used to detect error packets
X          *  sent by the receiver, in place of setjmp/longjmp
X          *  rdchk(fdes) returns non 0 if a character is available
X          */
X         flush_modem();
X         while (Bconstat(1)) {
X              switch (sreadline(1)) {
X              case CAN:
X              case ZPAD:
X                   zsdata(secbuf, 0, ZCRCE);
X                   goto waitack;
X              }
X         }
X#endif
X    } while (c == Blklen);
X    tcount += c;
X    lreport(tcount);
X    siggi = 0;
X    lsct = 1;
X
X    for (;;) {
X         stohdr(Txpos);
X         zsbhdr(ZEOF, Txhdr);
X         switch (getinsync()) {
```

```
X         case ZACK:
X              continue;
X         case ZRPOS:
X              goto somemore;
X         case ZRINIT:
X              return OK;
X         case ZSKIP:
X#ifndef REMOTE
X              fprintf(STDERR,"\r\nReceiver forced SKIP Transfer(2)\n\n");
X#endif
X              stfclose(in);
X              in = (-1);
X              return c;
X         default:
X              fprintf(STDERR,"\r\nErrors while Send Data\n\n");
X              stfclose(in);
X              in = (-1);
X              return ERROR;
X         }
X    }
X}
X
X/*
X * Respond to receiver's complaint, get back in sync with receiver
X */
Xgetinsync()
X{
X    register int c;
X
X    for (;;) {
X#ifdef TESTATTN
X         if (Testattn) {
X              wr_modem("\r\n\n\n***** Signal Caught *****\r\n");
X              Rxpos = 0; c = ZRPOS;
X         } else
X#endif
X
X              c = zgethdr(Rxhdr, 0);
X         switch (c) {
X         case ZCAN:
X         case ZABORT:
X         case ZFIN:
X         case TIMEOUT:
X              return ERROR;
X         case ZRPOS:
X              if (Lastc >= 0 && Lastread == Rxpos) {
X                   Dontread = TRUE;
X              } else {
X                   /* clearerr(in); */    /* In case file EOF seen */
X                   if(stfseek(in, Rxpos, 0))
X                   {
X                        fprintf(STDERR,"\r\nError While Seeking file\n");
X                        return ERROR;
```

```
X                  }
X
X             }
X             Txpos = Rxpos;
X             return c;
X         case ZACK:
X             return c;
X         case ZRINIT:
X         case ZSKIP:
X#ifndef REMOTE
X             fprintf(STDERR,"\n\n");
X#endif
X             stfclose(in);
X             in = (-1);
X             return c;
X         case ERROR:
X         default:
X             zsbhdr(ZNAK, Txhdr);
X             continue;
X         }
X    }
X}
X/* Say "bibi" to the receiver, try to do it cleanly */
Xsaybibi()
X{
X    for (;;) {
X         stohdr(0L);
X         zsbhdr(ZFIN, Txhdr);
X         switch (zgethdr(Rxhdr, 0)) {
X         case ZFIN:
X             sendline('O'); sendline('O'); flush_modem();
X         case ZCAN:
X         case TIMEOUT:
X             return;
X         }
X    }
X}
X
X/* Send command and related info */
Xzsendcmd(buf, blen)
Xchar *buf;
X{
X    register int c, errors;
X    long cmdnum;
X
X/*  cmdnum = getpid(); */
X    cmdnum = 1;   /* A random # */
X    errors = 0;
X    for (;;) {
X         stohdr(cmdnum);
X         Txhdr[ZF0] = Cmdack1;
X         zsbhdr(ZCOMMAND, Txhdr);
X         zsdata(buf, blen, ZCRCW);
```

```
Xlisten:
X         Rxtimeout = 100;        /* Ten second wait for resp. */
X         c = zgethdr(Rxhdr, 1);
X
X         switch (c) {
X         case ZRINIT:
X              continue;
X         case ERROR:
X         case TIMEOUT:
X              if (++errors > Cmdtries)
X                   return ERROR;
X              continue;
X         case ZCAN:
X         case ZABORT:
X         case ZFIN:
X         case ZSKIP:
X         case ZRPOS:
X              return ERROR;
X         default:
X              if (++errors > 10)
X                   return ERROR;
X              continue;
X         case ZCOMPL:
X              Exitcode = Rxpos;
X              saybibi();
X              return OK;
X         case ZRQINIT:
X              vfile("******** RZ *******");
X/*            stsystem("rz"); */
X              vfile("******** SZ *******");
X              goto listen;
X         }
X    }
X}
X
X
X#ifndef STANDALONE
X/*
X * If called as sb use YMODEM protocol
X */
Xschkinvok(s)
Xchar *s;
X{
X    if (s[0]=='s' && s[1]=='b') {
X         Nozmodem = TRUE; Blklen=KSIZE;
X    }
X}
X#endif
X
X/* -eof- */
SHAR_EOF
echo "File SZ.C is complete"
chmod 0600 SZ.C || echo "restore of SZ.C fails"
```

```
echo "x - extracting SZLNK (Text)"
sed 's/^X//' << 'SHAR_EOF' > SZLNK &&
Xc:\lib\gemstart.o common.o sz.o util.o tyme.o zm.o fileio.o
Xc:\lib\osbind.o c:\lib\gemlib c:\lib\libf
SHAR_EOF
chmod 0600 SZLNK || echo "restore of SZLNK fails"
echo "x - extracting TRANSFER.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > TRANSFER.C &&
X/*
X *  (Quick & Dirty) Transfer Shell
X *
X *      Jwahar Bammi
X *           usenet: mandrill!bammi@{decvax,sun}.UUCP
X *           csnet:  bammi@mandrill.ces.CWRU.edu
X *           arpa:   bammi@mandrill.ces.CWRU.edu
X *           CompuServe: 71515,155
X */
X
X#include "config.h"
X
X#ifdef DLIBS
X#include <string.h>
X#endif
X
X#include "zmdm.h"
X#include "common.h"
X
X#define ISWILD(X) ((X == '*')||(X == '?'))
X#define PROMPT        fprintf(STDERR,"zmdm> ")
X
Xextern int dorz(), dosz(), ls(), rm(), cp(), cd(), md(), rd(), pwd(), df(),
X       hhelp();
X
X#ifdef RECURSE
Xextern int doszf();
X#endif
X
Xstatic struct comnds {
X    char *command; /* command string  */
X    int     (*routine)();  /* routine to invoke    */
X    char *synopsis;    /* synopsis        */
X    int  expand;       /* expand wildcards before calling routine? */
X} comtab [] = {
X    { "rz",  dorz, "receive files using Z/X modem protocol",       TRUE },
X    { "rb", dorz, "receive files using Y modem protocol",         TRUE },
X#ifdef RECURSE
X    { "sz", doszf, "send files using Z/Y/X modem protocol",        TRUE },
X#else
X    { "sz", dosz, "send files using Z/Y/X modem protocol",        TRUE },
X#endif
X    { "sb", dosz, "send files using Y modem protocol",            TRUE },
X    { "rm", rm,   "remove files",                                 TRUE },
X    { "cp", cp,   "copy files",                                   TRUE },
```

```
X    { "ls", ls,   "list directory",                                  FALSE},
X    { "cd", cd,   "change working directory",           FALSE},
X    { "md", md,   "make a directory",                   FALSE},
X    { "rd", rd,   "remove a directory",                    FALSE},
X    { "pwd",pwd,  "print  working directory",           FALSE},
X    { "df", df,   "check free space",                   FALSE},
X    { "?",  hhelp, "this message",                      FALSE},
X    { (char *)NULL, (int (*)())NULL, (char *)NULL,                   FALSE}
X};
X
X#define MAXARGS   1024
Xstatic char *targv[MAXARGS];
Xstatic int targc;
Xchar *alltolower();
X
Xtransfer()
X{
X    char linebuf[132];
X    char *line;
X    int command;
X    int status;
X    extern int find_command();
X    extern int expnd_args();
X#ifdef DEBUG
X    int i;
X#endif
X
X    fprintf(STDERR,"hit <RETURN> to return to emulator,  <?> for help\n\n");
X    targc = 0;
X    while (TRUE)
X    {
X        if(targc > 1)
X            free_args();
X
X        PROMPT;
X        linebuf[0] = 127;
X#ifndef REMOTE
X        Cconrs(linebuf);
X#else
X        Cconraux(linebuf);
X#endif
X        putc('\n', STDERR);
X        if(linebuf[1] == 0)
X        /* cancelled */
X        return;
X
X        linebuf[(linebuf[1]+2)] = '\0';
X        line = &linebuf[2];
X#ifdef DEBUG
Xprintf("Line: |%s|\n", line);
X#endif
X
X        targv[0] = line;
```

```
X         targc    = 1;
X
X         /* pick up targv[0] */
X         while((*line != '\0') && (!isspace(*line)))
X              line++;
X
X         if(*line != '\0')
X         {
X              *line++ = '\0';
X         }
X
X         if((command = find_command(targv[0])) < 0)
X         {
X              fprintf(STDERR,"Invalid Command\n");
X              continue;
X         }
X
X         if(expnd_args(line, comtab[command].expand))
X              /* too many args */
X              continue;
X#ifdef DEBUG
Xprintf("targc %d\n", targc);
Xfor(i = 0; i < targc; i++)
X    printf("%s ", targv[i]);
Xprintf("\n\n");
X#endif
X
X         if((status = (*(comtab[command].routine))(targc, targv)) != 0)
X              fprintf(STDERR,"Exit Status %d\n", status);
X
X#ifdef DEBUG
Xprintf("Exit Status %d\n", status);
X#endif
X
X    } /* While */
X}
X
X/*
X * Straight sequential search thru comtab
X */
Xint find_command(s)
Xregister char *s;
X{
X    register int i;
X
X    for(i = 0; comtab[i].command != (char *)NULL; i++)
X    {
X         if(strcmp(s, comtab[i].command) == 0)
X              return i;
X    }
X
X    return -1;
X}
```

```
X
X/*
X * Expand command line args, return TRUE if too many args, or Not matching Quotes
X */
Xint expnd_args(s, expand_wild)
Xregister char *s;
Xint expand_wild;
X{
X    char next_arg[128];
X    register char *p;
X    register int contains_wild;
X    extern int add_argv();
X    extern int handl_wild();
X
X    while(*s != '\0')
X    {
X         p = next_arg;
X         while(isspace(*s)) s++; /* skip leading space */
X         if(*s != '\0')
X         {
X              contains_wild = FALSE;
X              if(*s == '\047')
X              {
X                   /* Quoted arg */
X                   s++;
X                   while((*s != '\0') && (*s != '\047'))
X                        *p++ = *s++;
X                   *p = '\0';
X                   if(*s == '\0')
X                   {
X                        fprintf(STDERR,"No Matching Quote\n");
X                        return TRUE;
X                   }
X                   else
X                        s++;
X                   if(add_argv(next_arg))
X                        return TRUE;
X              }
X              else
X              {
X                   while(!isspace(*s) && (*s != '\0'))
X                   {
X                        contains_wild |= ISWILD(*s);
X                        *p++ = *s++;
X                   }
X                   *p = '\0';
X
X                   if(contains_wild && expand_wild)
X                   {
X                        if(handl_wild(next_arg))
X                             return TRUE;
X                   }
X                   else
```

```
X                  {
X                       if(add_argv(next_arg))
X                            return TRUE;
X                  }
X             } /* if-else */
X         } /* If */
X    } /* while */
X
X    return FALSE;
X}
X
X/*
X * add an arg to argv. Return TRUE if error
X */
Xint add_argv(s)
Xchar *s;
X{
X    extern char *myalloc();
X    extern char *strcpy();
X    extern int strlen();
X
X    if(targc > (MAXARGS-1))
X    {
X         fprintf(STDERR,"Too many arguements (%d Max)\n", MAXARGS);
X         return TRUE;
X    }
X    targv[targc++] = strcpy(myalloc(strlen(s)+1), s);
X
X    return FALSE;
X
X}
X
X/*
X * expand wild card arguments. Return TRUE on error.
X */
Xint handl_wild(s)
Xchar *s;
X{
X    extern struct stat statbuf;
X    extern char *mkpathname();
X
X    if(Fsfirst(s, 0x21) != 0)
X    {
X         /* No match */
X         fprintf(STDERR,"No Match for %s\n", s);
X         return TRUE;
X    }
X
X    alltolower(statbuf.st_name);
X    if(add_argv(mkpathname(s, statbuf.st_name)))
X         return TRUE;
X
X    while(Fsnext() == 0)
```

```
X    {
X         alltolower(statbuf.st_name);
X         if(add_argv(mkpathname(s, statbuf.st_name)))
X              return TRUE;
X    }
X
X    return FALSE;
X}
X
X/*
X * Given a spec with a trailing wildcard and a base will name construct pathname
X *
X */
Xchar *mkpathname(spec, file)
Xregister char *spec, *file;
X{
X    extern char *rindex();
X    register char *p;
X
X    if((p = rindex(spec, '\\')) == (char *)NULL)
X         /* no path name */
X         return file;
X
X    while(*file != '\0')
X         *++p = *file++;
X    *++p = '\0';
X
X    return spec;
X}
X
Xfree_args()
X{
X    register int i;
X
X    for(i = 1; i < targc; i++)
X         free(targv[i]);
X}
X
X/*
X * remove files
X *  Usage: rm [-i] files ...
X */
Xint rm(argc, argv)
Xregister int argc;
Xregister char **argv;
X{
X    register int interactive;
X    register int status, s;
X    extern int yesno();
X    extern int errno;
X
X    interactive = FALSE;
X    status = 0;
```

```
X    while(--argc)
X    {
X         ++argv;
X         if( ((*argv)[0] == '-') && ((*argv)[1] == 'i') )
X              interactive = TRUE;
X         else
X         {
X              if(interactive)
X                   if(!yesno("rm: remove", *argv))
X                        continue;
X              if(unlink(*argv))
X              {
X                   s |= 1;
X                   fprintf(STDERR, "%s: no such file\n", *argv);
X              }
X              status |= s;
X         }
X    }
X    return status;
X}
X
X/*
X * Prompt and return Yes/No truth value
X *
X */
Xint yesno(p1, p2)
Xregister char *p1, *p2;
X{
X    char reply[16];
X
X    fprintf(STDERR,"%s %s (y/n): ", p1, p2);
X    reply[0] = 16;
X#ifndef REMOTE
X    Cconrs(reply);
X#else
X    Cconraux(reply);
X#endif
X
X    putc('\n', STDERR);
X    return ( (reply[2] == 'y') || (reply[2] == 'Y') );
X
X}
X
X/*
X * copy files
X *  Usage:
X *       cp src dest
X *      or
X *       cp files.. directory
X */
Xint cp(argc, argv)
Xint argc;
Xchar **argv;
```

```
X{
X    char dest[128];
X    register int status;
X    extern int strlen();
X    extern int cpy();
X    extern char *basename();
X
X    status = 0;
X    if(argc > 3)
X    {
X         register int i;
X
X         if(!existd(argv[argc-1]))
X         {
X              fprintf(STDERR,"%s does not exists or is not a directory\n",
X                   argv[argc-1]);
X              return 1;
X         }
X
X         for(i = 1; i < argc - 1; i++)
X         {
X              strcpy(dest, argv[argc-1]);
X              if( (argv[argc-1])[(strlen(argv[argc-1])-1)] != '\\')
X                   strcat(dest, "\\");
X              strcat(dest, basename(argv[i]));
X
X              fprintf(STDERR,"copying %s to %s\n", argv[i], dest);
X              status |= cpy(argv[i], dest);
X         }
X    }
X    else
X    {
X         if(argc > 2)
X         {
X              if(existd(argv[2]))
X              {
X                   /* dest is a directory */
X                   strcpy(dest, argv[2]);
X                   if( (argv[2])[(strlen(argv[2])-1)] != '\\')
X                        strcat(dest, "\\");
X
X                   strcat(dest, basename(argv[1]));
X
X                   fprintf(STDERR,"copying %s to %s\n", argv[1], dest);
X                   return (cpy(argv[1], dest));
X              }
X
X              if(strcmp(argv[1], argv[2]) == 0)
X              {
X                   fprintf(STDERR,"Cannot copy a file onto itself\n");
X                   return 3;
X              }
X              status = cpy(argv[1], argv[2]);
```

```
X         }
X         else
X         {
X              fprintf(STDERR,"Usage: cp source dest or cp files .. directory\n");
X              return 2;
X         }
X    }
X
X    return status;
X}
X
X
X/*
X * Cpy src -> dest
X *
X */
Xint cpy(src, dest)
Xchar *src, *dest;
X{
X    register int fps,fpd;
X    register long count;
X    register int status;
X
X    status = 0;
X
X    if((fps = Fopen(src, 0)) < (-3))
X    {
X         status = fps;
X         fprintf(STDERR,"%s: no such file\n", src);
X         return status;
X    }
X
X    if((fpd = Fcreate(dest, 0)) < (-3))
X    {
X         if((fpd = Fopen(dest, 1)) < (-3))
X         {
X              Fclose(fps);
X              status = fpd;
X              fprintf(STDERR,"%s: cannot create\n",dest);
X              return status;
X         }
X    }
X
X    while( (count = Fread(fps, (long)BBUFSIZ, bufr)) > 0L)
X    {
X         if(Fwrite(fpd, count, bufr) != count)
X         {
X              status = 1;
X              fprintf(STDERR,"Error Writing %s\n",dest);
X              break;
X         }
X    }
X    if(count < 0L)
```

```
X    {
X         status = 2;
X         fprintf(STDERR,"Error Reading %s\n", src);
X    }
X
X    Fclose(fpd);
X    Fclose(fps);
X
X    return status;
X}
X
X#define haswild(X) \
X( (rindex(X,'*') != (char *)NULL) || (rindex(X,'?') != (char *)NULL) )
X
X/*
X * list directories
X */
Xint ls(argc, argv)
Xint argc;
Xchar **argv;
X{
X    char path[128];
X    register char *p;
X    extern char *rindex();
X    extern int existd();
X
X    if(argc < 2)
X         lis("*.*");
X    else
X    {
X         while(--argc)
X         {
X              ++argv;
X              if((p = rindex(*argv,'\\')) == (char *)NULL)
X                   p = *argv;
X              else
X                   p++;
X              if(*p == '\0')
X              {
X                   strcpy(path, *argv);
X                   strcat(path,"*.*");
X              }
X              else
X              {
X                   if(haswild(p))
X                        strcpy(path, *argv);
X                   else
X                   {
X                        if(existd(p))
X                        {
X                             strcpy(path, *argv);
X                             strcat(path, "\\*.*");
X                        }
```

```
X                       else
X                            strcpy(path, *argv);
X                   }
X              }
X              lis(path);
X         } /* while */
X    }
X
X    return 0;
X}
X
X/*
X * given a possibly wild carded string put out list of subtrees
X *
X */
Xlis(wild)
Xchar *wild;
X{
X    extern struct stat statbuf;
X    register int count;
X
X#ifdef DEBUG
Xprintf("ls %s\n", wild);
X#endif
X
X    if(Fsfirst(wild, 0x0020 | 0x0010) != 0)
X    {
X         fprintf(STDERR,"%s - no match.\n", wild);
X         return;
X    }
X
X    count = 0;
X    alltolower(statbuf.st_name);
X    if(!((strcmp(statbuf.st_name,".") == 0) ||
X       (strcmp(statbuf.st_name, "..") == 0)))
X         putls(&statbuf, ++count);
X
X    while(Fsnext() == 0)
X    {
X         alltolower(statbuf.st_name);
X         if(!((strcmp(statbuf.st_name,".") == 0) ||
X            (strcmp(statbuf.st_name, "..") == 0)))
X              putls(&statbuf, ++count);
X    }
X
X    if((count % 4))
X         putc('\n', STDERR);
X
X}
X
X/*
X * Put out a directory entry
X */
```

```
Xputls(statbuf, count)
Xregister struct stat *statbuf;
Xregister int count;
X{
X    char dbuf[16];
X
X    if((statbuf->st_mode) & 0x0010)
X    {
X         /* subtree */
X         strcpy(dbuf, statbuf->st_name);
X         strcat(dbuf, "/");
X         fprintf(STDERR,"%-13s      ", dbuf);
X    }
X    else
X         /* file */
X         fprintf(STDERR,"%-12s %5ld ", statbuf->st_name, statbuf->st_size);
X
X    if(!(count %4))
X         putc('\n', STDERR);
X}
X
X/*
X * Change working directory
X */
Xint cd(argc, argv)
Xint argc;
Xchar **argv;
X{
X    register char *d, *path;
X    register int drive;
X    extern char *index();
X
X    if((argc < 2) || (argc > 2))
X    {
X         fprintf(STDERR,"Usage: cd directory\n");
X         return 1;
X    }
X
X
X
X    if((d = index(argv[1],':')) == (char *)NULL)
X    {
X         /* Drive was not specified, must mean the current drive */
X         path = argv[1];
X    }
X    else
X    {
X         d--;
X         if(isupper(*d))
X         {
X              *d = tolower(*d);
X         }
X         drive = *d - 'a';
```

```
X         if(d[2] != '\\')   /* just gave D: */
X         {
X              /* we will shove in the '\' */
X              d[1] = '\\';
X              path = &d[1];
X         }
X         else
X              path = &d[2];
X
X         /* Set the Drive */
X         if(Dsetdrv(drive) < 0)
X         {
X              fprintf(STDERR,"Could not set drive %c:\n", *d);
X              return 2;
X         }
X
X    }
X
X
X    /* Set the Path */
X    if(Dsetpath(path) != 0)
X    {
X         fprintf(STDERR,"Could not set directory %s\n", path);
X         return 3;
X    }
X
X    return 0;
X
X}
X
X/*
X * pwd - figure out the current working directory
X *
X */
Xint pwd()
X{
X    register int drive;
X    char _cwd[128];
X
X    drive = Dgetdrv();
X    Dgetpath(&_cwd[3],0);
X    fprintf(STDERR,"%c:%s%s\n", (drive + 'a'),
X         ((_cwd[3] == '\\')? "" : "\\"), &_cwd[3]);
X
X    return 0;
X}
X
X/*
X * Print free space
X */
Xint df(argc, argv)
Xint argc;
Xchar **argv;
```

```
X{
X    struct {
X        long b_free;
X        long b_total;
X        long b_secsiz;
X        long b_clsiz;
X    } fbuf;
X
X    register int drive;
X    register long ffree, total;
X    extern long drv_map;
X
X    if(argc > 1)
X    {
X         ++argv;
X         drive = (*argv)[0];
X         if(isupper(drive))
X              drive = tolower(drive);
X
X         drive = drive - 'a';
X         if((drv_map & (1L << drive)) == 0)
X         {
X              fprintf(STDERR,"Invalid Drive %c:\n", (drive + 'a'));
X              return 1;
X         }
X         drive +=1;
X    }
X    else
X         drive = Dgetdrv()+1;    /* default current drive */
X
X    fprintf(STDERR,"Please Wait .....");
X    Dfree(&fbuf,drive);
X    ffree = fbuf.b_free * fbuf.b_clsiz * fbuf.b_secsiz;
X    total =  fbuf.b_total * fbuf.b_clsiz * fbuf.b_secsiz;
X    fprintf(STDERR,"\r%ld Bytes (", ffree);
X    prratio(STDERR, ffree, total);
X    fprintf(STDERR, ") Free on drive %c:\n", ((drive-1) + 'a'));
X
X    return 0;
X}
X
X
X/*
X * print a ratio
X * avoid floats like the plague
X */
Xprratio(stream, num, den)
XFILE *stream;
Xlong  num, den;
X{
X        register int q;                 /* Doesn't need to be long */
X
X        if(num > 214748L) {             /* 2147483647/10000 */
```

```
X                q = num / (den / 10000L);
X        } else {
X                q = 10000L * num / den;         /* Long calculations, though */
X        }
X        if (q < 0) {
X                putc('-', stream);
X                q = -q;
X        }
X        fprintf(stream, "%d.%02d%%", q / 100, q % 100);
X}
X
X/*
X * Make directories
X */
Xint md(argc, argv)
Xint argc;
Xchar **argv;
X{
X    register int status, s;
X
X    status = 0;
X    while(--argc)
X    {
X         ++argv;
X         if(( s = Dcreate(*argv)))
X         {
X              status |= s;
X              fprintf(STDERR,"Could not create %s\n", *argv);
X         }
X    }
X
X    return status;
X}
X
X/*
X * Remove directories
X */
Xint rd(argc, argv)
Xint argc;
Xchar **argv;
X{
X    register int status, s;
X
X    status = 0;
X    while(--argc)
X    {
X         ++argv;
X         if(( s = Ddelete(*argv)))
X         {
X              status |= s;
X              fprintf(STDERR,"Could not delete %s\n", *argv);
X         }
X    }
```

```
X
X    return status;
X}
X
X
X/*
X * Allocate memory with error check
X *
X */
Xchar *myalloc(size)
Xunsigned int size;
X{
X    register char *m;
X    extern char *malloc();
X
X    if((m = malloc(size)) == (char *)NULL)
X    {
X         fprintf(STDERR,"Out of Memory\nSorry, cannot continue\n");
X         exit(1);
X    }
X
X    return m;
X}
X
X/*
X * Put out help
X */
Xint hhelp()
X{
X    register int i;
X
X#ifdef REMOTE
X    fprintf(STDERR,"\n\t\t\t    Available Commands \n\n");
X#else
X    fprintf(STDERR,"\n\t\t\t   \033p Available Commands \033q\n\n");
X#endif
X    for(i = 0; comtab[i].command != (char *)NULL; i++)
X    {
X         fprintf(STDERR,"\t%s\t%s\n", comtab[i].command,
X              comtab[i].synopsis);
X    }
X    fprintf(STDERR,"\n\tor hit <RETURN> to exit to terminal emulator\n\n");
X    return 0;
X
X}
X
X/*
X * Lower case string
X */
Xchar *alltolower(s)
Xregister char *s;
X{
X    register char *p;
```

```
X
X    for(p = s; *p != '\0'; p++)
X    {
X         if(isupper(*p))
X              *p = tolower(*p);
X    }
X    return s;
X}
X
X/*
X * return the base filename, given (potentially) a pathname
X */
Xchar *basename(s)
Xregister char *s;
X{
X    register char *p;
X    extern char *rindex();
X
X    if((p = rindex(s, '\\')) == (char *)NULL)
X    {
X         if((p = rindex(s, ':')) == (char *)NULL)
X              return s;
X         else
X              return ++p;
X    }
X    return ++p;
X}
X
X#ifdef RECURSE
Xint doszf(argc, argv)
Xint argc;
Xchar **argv;
X{
X    extern int expandargs();
X#ifdef DDEBUG
Xint i;
Xprintf("doszf: argc %d\n", argc);
Xfor(i = 0; i < argc; i++)
X    printf("%s ", argv[i]);
Xprintf("\n\n");
X#endif
X
X    return(expandargs(dosz, argc, argv));
X}
X
X#include "expandar.c"
X
X#endif
X
X#ifdef REMOTE
X/*
X * Read a line from the Aux port
X *
```

```
X */
X
XCconraux(l)
Xchar *l;
X{
X    register int c;
X    register int count;
X    register char *p;
X
X    p = &l[2];
X    count = 0;
X
X    while((c = get_a_c()) != '\n')
X    {
X         *p++ = c;
X         count++;
X    }
X    l[1] = count;
X}
X
Xstatic char _chr_buf[128];
Xstatic int _n_chr = 0;
Xstatic char *_next_chr = &_chr_buf[0];
Xstatic char *_next_avail = &_chr_buf[0];
X
Xint get_a_c()
X{
X        register int c;
X
X        if(_n_chr > 0)
X        {
X                _n_chr--;
X                return(*_next_avail++);
X        }
X
X        _n_chr = 0;
X        _next_avail = &_chr_buf[0];
X        _next_chr = &_chr_buf[0];
X
X        while( ((c = Bconin(1) & 0x7fff) != '\r') && (c != '\n'))
X        {
X                if((c == '\003') || (c == '\025'))
X                {
X             while(_n_chr-- > 0)
X                  wr_modem("\b \b");
X                        _n_chr = 0;
X                        _next_avail = &_chr_buf[0];
X                        _next_chr = &_chr_buf[0];
X
X                }
X
X                if((c == '\b') || (c == '\177'))
X                {
```

```
X                       if(_n_chr > 0)
X                       {
X                               _next_chr--;
X                               _n_chr--;
X                               wr_modem("\b \b");
X                       }
X               }
X               else
X               {
X                       *_next_chr++ = c;
X                       _n_chr++;
X             Bconout(1, c);
X               }
X        }
X
X        *_next_chr = '\n';
X        return(*_next_avail++);
X}
X#endif /* REMOTE */
X
X/* -eof- */
SHAR_EOF
chmod 0600 TRANSFER.C || echo "restore of TRANSFER.C fails"
echo "x - extracting TYME.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > TYME.C &&
X/*
X *  Time conversions Module
X *
X *      Jwahar Bammi
X *           usenet: mandrill!bammi@{decvax,sun}.UUCP
X *           csnet:  bammi@mandrill.ces.CWRU.edu
X *           arpa:   bammi@mandrill.ces.CWRU.edu
X *           CompuServe: 71515,155
X */
X
X#include "config.h"
X#include <stdio.h>
X
X
X/*
X * days in a given year
X */
X#define days_in_year(Y) (leap(Y) ? 366 : 365)
X
X/* # of days / month in a normal year */
Xstatic unsigned int md[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
X
Xstatic int leap (y)
Xint y;
X{
X    y += 1900;
X    if ((y % 400) == 0)
X        return (1);
```

```
X    if ((y % 100) == 0)
X        return (0);
X    return ((y % 4) == 0);
X}
X
X/* Return the number of days between Jan 1, Given Year and the given
X * broken-down time.
X */
X
Xstatic unsigned long ndays (since, year, month, day)
Xunsigned int year, month, day;
X{
X    register unsigned long n = day;
X    register unsigned int m, y;
X
X    for (y = since; y < year; y++)
X    {
X         n += 365;
X         if (leap (y)) n++;
X    }
X    for (m = 0; m < (month-1); m++)
X        n += md[m] + ( ((m == 1) && leap(y))? 1 : 0);
X
X    return (n);
X}
X
X/* Convert a broken-down time into seconds
X *
X */
X
Xunsigned long tm_to_time (base_year, year, month, day, hours, mins, secs)
Xunsigned int base_year, year, month, day, hours, mins, secs;
X{
X    register unsigned long t;
X    extern unsigned long ndays();
X
X    t = (ndays(base_year, year, month, day) - 1L) * (unsigned long)86400L
X        + hours * (unsigned long)3600L + mins * (unsigned long)60L + secs;
X
X    return t;
X}
X
X/*
X * Convert ST time to Unix time
X *
X */
Xunsigned long st2unix(time, date)
Xunsigned int time, date;
X{
X    extern unsigned long tm_to_time();
X
X    unsigned int yr = ((date >> 9) & 0x007f) + 80;  /* dissect the date */
X    unsigned int mo = (date >> 5) & 0x000f;
```

```
X    unsigned int dy = date & 0x1f;
X
X    unsigned int hr = (time >> 11) & 0x001f;        /* dissect the time */
X    unsigned int mm = (time >> 5)  & 0x003f;
X    unsigned int ss = (time & 0x001f) * 2;
X
X#ifdef SDEBUG
X    printf("%d/%d/%d  %d:%d:%d\n", mo, dy, yr,hr, mm, ss);
X#endif
X    return (tm_to_time(70, yr, mo, dy, hr, mm, ss) -
X         (unsigned long)GMTDIFF);
X}
X
X/*
X * Convert Unix Time to ST Time
X *
X */
Xvoid unix2st(Unix, time, date)
Xunsigned long Unix;
Xunsigned int *date, *time;
X{
X    long stbase;
X    unsigned int hours, yrs, day, months, mins, seconds, t;
X    long days, secs;
X    extern unsigned long tm_to_time();
X
X#ifdef DEBUG
Xprintf("\n\nUnix Time %ld\n", Unix);
X#endif
X
X
X    if((Unix - tm_to_time(70, 80, 0, 0, 0, 0, 0)) <= 0)  /* base 1980 */
X    {
X         /* thats before St's time */
X         *time = 0;
X         *date = (1 << 5) | 1;   /* Jan 1, 1980 00:00:00 GMT */
X#ifdef DEBUG
Xprintf("Before my time\n");
X#endif
X         return;
X    }
X
X    stbase = Unix;     /* do from base year 1970 */
X
X    days = stbase / 86400L;/* 3600*24 */
X    secs = stbase % 86400L + GMTDIFF;
X    if(secs < 0)  /* previous day here */
X    {
X         days -= 1;
X         secs += 86400L;
X    }
X
X    /* extract hrs : mins : seconds */
```

```
X    hours = secs / 3600;
X    secs = secs - (hours * 3600);
X    mins = secs / 60;
X    seconds = secs - (mins * 60);
X    seconds &= ~1L;             /* ST has 2 sec resolution */
X
X    /* get the year and day of the year */
X    for(t = 70; days >= days_in_year(t); t++)
X         days -= days_in_year(t);
X    yrs = t;
X    day = days;
X
X    /* get the month */
X    if(days_in_year(yrs) == 366)
X         md[1] = 29;
X
X    for(t = 0; day >= md[t]; t++)
X         day -= md[t];
X
X    md[1] = 28;
X    day = day + 1;
X    months = t + 1;
X
X#ifdef DEBUG
Xprintf("%d/%d/%d   %d:%d:%d\n", months, day, yrs, hours, mins, seconds);
X#endif
X
X    yrs -= 80;
X    *date = (((yrs & 0x007f) << 9) | ((months & 0x000f) << 5)
X          | (day & 0x001f));
X
X    *time = (((hours & 0x001f) << 11) | ((mins & 0x003f) << 5)
X          | (seconds & 0x001e));
X}
X
X#ifdef TEST
X#include <stdio.h>
X#include <osbind.h>
X
Xmain()
X{
X    unsigned int time, date;
X    unsigned long Unix;
X    extern unsigned long st2unix();
X
X    time = Tgettime();
X    date = Tgetdate();
X    Unix = st2unix(time, date);
X
X    printd(time, date);
X    printf("Unix Time %ld\n", Unix);
X
X    unix2st(Unix, &time, &date);
```

```
X    printd(time, date);
X}
X
Xprintd(time, date)
Xunsigned int time, date;
X{
X
X    unsigned int yr = ((date >> 9) & 0x007f) + 80;  /* dissect the date */
X    unsigned int mo = (date >> 5) & 0x000f;
X    unsigned int dy = date & 0x1f;
X
X    unsigned int hr = (time >> 11) & 0x001f;        /* dissect the time */
X    unsigned int mm = (time >> 5)  & 0x003f;
X    unsigned int ss = (time & 0x001f) * 2;
X
X    printf("%d/%d/%d\t%d:%d:%d\n", mo, dy, yr, hr, mm, ss);
X}
X#endif /* TEST */
X
X/* -eof- */
SHAR_EOF
chmod 0600 TYME.C || echo "restore of TYME.C fails"
echo "x - extracting UTIL.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > UTIL.C &&
X/*
X *  Utilities Module
X *
X *       Jwahar Bammi
X *            usenet: mandrill!bammi@{decvax,sun}.UUCP
X *            csnet:  bammi@mandrill.ces.CWRU.edu
X *            arpa:   bammi@mandrill.ces.CWRU.edu
X *            CompuServe: 71515,155
X */
X
X
X#include "config.h"
X#include "zmdm.h"
X
X#ifndef Vsync              /* Atari forgot these in osbind.h */
X#define Vsync()   xbios(37)
X#endif
X
X#define GOTO(r,c) EscSeq('Y');Bconout(2, r+040);Bconout(2, c+040)
X
X    /* External Variables */
Xlong     pr_time   = 0L;     /* Present time    (200 Hz) */
X
X#ifndef STANDALONE
X#ifndef REMOTE
Xextern int  *aline_addr; /* Ptr to base of Aline variables */
Xextern long *ms_ptr;  /* pointer to my screen memory aligned to a 256
X                byte boundary */
X
```

```
X    /* Globals */
Xstatic char *hs_ptr;   /* pointer to his screen memory */
Xstatic int  x_saved, y_saved;   /* Saved cursor position */
X#endif
X#endif
X
Xstatic long *hz_200 =  (long *)0x000004ba; /* Yes the Hitch Hikers */
X                         /* Guide is wrong!! */
X
X/*
X * rd_time() - read the value of the systems 200 Hz counter
X * must be called in Super Mode else you get
X * what you deserve - MUSHROOMS!!
X */
Xvoid rd_time()
X{
X    pr_time = *hz_200;
X}
X
X#ifndef STANDALONE
X/*
X * my_screen() - Start using my screen memory
X * for all output.
X *
X */
Xvoid my_screen()
X{
X#ifndef REMOTE
X    /* The cursor position has been saved prior to calling this routine */
X    x_saved = aline_addr[-14];
X    y_saved = aline_addr[-13];
X
X    /* save his screen memory pointer */
X    hs_ptr = (char *)Logbase();
X
X    /* switch to my display memory */
X    Setscreen(ms_ptr, ms_ptr, -1);
X#else
X    wr_modem("\r\n\n");
X#endif
X}
X
X
Xvoid his_screen()
X{
X#ifndef REMOTE
X    /* switch to his Screen memory, wait for a Vblank, so
X     * that the Logical Loc == Physical Loc, then pop saved
X     * cursor */
X    Setscreen(hs_ptr, hs_ptr, -1);
X    Vsync();
X    Vsync(); /* starts happening at the last one */
X    GOTO(y_saved, x_saved);
```

```
X#else
X    wr_modem("\r\n\n");
X#endif
X
X}
X
X/*
X * hit_key() - wait for the user to hit a key
X *
X */
Xvoid hit_key()
X{
X#ifndef REMOTE
X    Bconws("\r\n\033pHit Any Key To Continue .....\033q");
X    Bconin(2);
X    Bconws("\r\n");
X#else
X    wr_modem("\r\nHit Any Key To Continue .....");
X    Bconin(1);
X    wr_modem("\r\n");
X#endif
X}
X#endif /* STANDALONE */
X
X/*
X * find the size of a file given its name
X */
Xlong filesize(name)
Xregister char *name;
X{
X    extern struct stat statbuf;
X
X    if(Fsfirst(name,(int)(0x01 | 0x020)) != 0)
X    {
X         return 0L;
X
X    }
X
X    return statbuf.st_size;
X}
X
X
X/*
X * Write a string to console
X */
XBconws(s)
Xregister char *s;
X{
X    while(*s)
X         Bconout(2, *s++);
X}
X
X#ifndef STANDALONE
```

```
X#ifndef REMOTE
X#ifndef MANX
X#ifndef MWC
X
X/* **WARNING** **WARNING** **WARNING** **WARNING** **WARNING** **WARNING** */
X/*                                                 */
X/*  ALCYON C dependant, this routine must return the base address of
        */
X/*  of the linea variable block                          */
X/*                                                 */
X/* **WARNING** **WARNING** **WARNING** **WARNING** **WARNING** **WARNING** */
X
X/*
X * return the base address of the line A variables
X */
Xint *aaddress()
X{
X    asm("dc.w $A000"); /* Line A trap - 0000 is init aline  */
X                  /* d0 and a0 now contain the address */
X                  /* so we can just return and the result
X                   * will be valid
X                   */
X}
X
X
X/*
X *  Make hi rez screen bios handle 50 lines of 8x8 characters
X *
X * Adapted to C use from origonal asm posting
X *
X */
Xhi50()
X{
X    /* Switch to 50 lines display */
X
X    asm(".dc.w    $a000");      /* get the important pointers */
X
X    asm("move.l  04(a1),a1");   /* a1 -> 8x8 font header */
X
X    asm("move.l  72(a1),-$0A(a0)");  /* v_off_ad <- 8x8 offset table addr */
X    asm("move.l  76(a1),-$16(a0)");  /* v_fnt_ad <- 8x8 font data addr */
X
X    asm("move.w  #008,-$2E(a0)");    /* v_cel_ht <- 8    8x8 cell height */
X    asm("move.w  #049,-$2A(a0)");    /* v_cel_my <- 49   maximum cell "Y" */
X    asm("move.w  #640,-$28(a0)"); /* v_cel_wr <- 640  offset to cell Y+1 */
X
X}
X
X/*
X * Make hi rez screen bios handle 25 lines of 8x16 characters
X *
X */
Xhi25()
```

```
X{
X
X    /* Switch to 25 lines display */
X
X
X    asm(".dc.w    $a000");      /* get the important pointers */
X
X    asm("move.l  08(a1),a1");   /* a1 -> 8x16 font header */
X
X    asm("move.l  72(a1),-$0A(a0)");  /* v_off_ad <- 8x16 offset table addr */
X    asm("move.l  76(a1),-$16(a0)");  /* v_fnt_ad <- 8x16 font data addr */
X
X    asm("move.w  #0016,-$2E(a0)");   /* v_cel_ht <- 16    8x16 cell height */
X    asm("move.w  #0024,-$2A(a0)");   /* v_cel_my <- 24    maximum cell "Y" */
X    asm("move.w  #1280,-$28(a0)");   /* v_cel_wr <- 1280  vertical byte offset */
X}
X
X#else
X
X/* **WARNING** **WARNING** **WARNING** **WARNING** **WARNING** **WARNING** */
X/*                                                 */
X/*     MWC  dependant, this routine must return the base address of
       */
X/*  of the linea variable block                          */
X/*                                                 */
X/* **WARNING** **WARNING** **WARNING** **WARNING** **WARNING** **WARNING** */
X#include <linea.h>
X/*
X * return the base address of the line A variables
X */
X
Xint *aaddress()
X{
X    linea0(); /* Init LineA - dumps returned values into la_init */
X    return (int *)(la_init.li_a0); /* Return address of Parameter Block */
X}
X
X        /* Mark Williams C        */
X        /* See file hi5025.s       */
X        /* for functions hi50()    */
X        /* and hi25()         */
X#endif /* MWC */
X#endif /* MANX */
SHAR_EOF
echo "End of part 5"
echo "File UTIL.C is continued in part 6"
echo "6" > s2_seq_.tmp
exit 0


From -788037400862656777
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,43c96a7bd1e64f43,start
```

X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-30 09:15:18 PST
Path: utzoo!attcan!uunet!husc6!mailrus!rutgers!ucsd!ucbvax!agate!saturn!ssyx!
koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i046: zmdm -- terminal program with file transfer part06/08
Keywords: mwc, alcyon, shar
Message-ID: <3512@saturn.ucsc.edu>
Date: 30 May 88 16:15:18 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1016
Approved: koreth@ssyx.ucsc.edu

Submitted-by: bammi@mandrill.ces.cwru.edu (Jwahar R. Bammi)
Posting-number: Volume 1, Issue 46
Archive-name: zmdm/part06

```
#!/bin/sh
# this is part 6 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file UTIL.C continued
#
CurArch=6
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file UTIL.C"
sed 's/^X//' << 'SHAR_EOF' >> UTIL.C
X#endif /* REMOTE */
X#endif /* STANDALONE */
X
X/* -eof- */
SHAR_EOF
echo "File UTIL.C is complete"
chmod 0600 UTIL.C || echo "restore of UTIL.C fails"
echo "x - extracting YMODEM.DOC (Text)"
sed 's/^X//' << 'SHAR_EOF' > YMODEM.DOC &&
X
X
X
X                                   - 1 -
X
X
X
X                     XMODEM/YMODEM PROTOCOL REFERENCE
X                 A compendium of documents describing the
```

X
X XMODEM and YMODEM
X
X File Transfer Protocols
X
X
X
X This document was formatted 9-11-86.
X
X
X
X
X
X
X Edited by Chuck Forsberg
X
X
X
X
X
X
X
X
X
X
X
X
X
X Please distribute as widely as possible.
X
X Questions to Chuck Forsberg
X
X
X
X
X
X Omen Technology Inc
X 17505-V Sauvie Island Road
X Portland Oregon 97231
X Voice: 503-621-3406
X Modem (Telegodzilla): 503-621-3746 Speed 1200,300
X Compuserve: 70007,2304
X UUCP: ...!tektronix!reed!omen!caf
X
X
X
X
X
X
X

X
X
X
X
X
X
X
X

X
X
X
X 1. ROSETTA STONE
X
X Here are some definitions which reflect the current vernacular
X in the computer media. The attempt here is identify the file
X transfer protocol rather than specific programs.
X
X XMODEM refers to the original 1979 file transfer etiquette
X introduced by Ward Christensen's 1979 MODEM2 program. It's also
X called the MODEM or MODEM2 protocol. Some who are unaware of MODEM7's
X unusual batch file mode call it MODEM7. Other aliases include "CP/M
X Users's Group" and "TERM II FTP 3". This protocol is supported by
X every serious communications program because of its universality,
X simplicity, and reasonable performance.
X
X XMODEM/CRC replaces XMODEM's 1 byte checksum with a two byte Cyclical
X Redundancy Check (CRC-16), giving modern error detection
X protection.
X
X XMODEM-1k Refers to the XMODEM/CRC protocol with 1024 byte data blocks.
X
X YMODEM refers to the XMODEM/CRC (optional 1k blocks) protocol with the
X batch transmission described below.
X
X ZMODEM uses familiar XMODEM/CRC and YMODEM technology in a
X new protocol that provides reliability, throughput, file
X management, and user amenities appropriate to contemporary
X data communications.
X
X
X 2. YET ANOTHER PROTOCOL?
X
X Since its development half a decade ago, the Ward Christensen modem
X protocol has enabled a wide variety of computer systems to interchange
X data. There is hardly a communications program that doesn't at least
X claim to support this protocol.
X
X Recent advances in computing, modems and networking have
X revealed a number of weaknesses in the original protocol:
X
X + The short block length caused throughput to suffer when used with
X timesharing systems, packet switched networks, satellite circuits,
X and buffered (error correcting) modems.

X
X + The 8 bit arithmetic checksum and other aspects allowed line
X impairments to interfere with dependable, accurate transfers.
X
X + Only one file could be sent per command. The file name had to be
X given twice, first to the sending program and then again to the
X receiving program.
X
X + The transmitted file could accumulate as many as 127 extraneous
X bytes.
X
X
X
X Chapter 2
X
X
X
X
X
X
X X/YMODEM Protocol Reference 09-11-86 3
X
X
X
X + The modification date of the file was lost.
X
X A number of other protocols have been developed over the
X years, but none have displaced XMODEM to date:
X
X + Lack of public domain documentation and example programs have kept
X proprietary protocols such as MNP, Blast, and others
X tightly bound to the fortunes of their suppliers.
X
X + Complexity discourages the widespread application of BISYNC, SDLC,
X HDLC, X.25, and X.PC protocols.
X
X + Performance compromises and moderate complexity have limited the
X popularity of the Kermit protocol, which was developed to allow file
X transfers in environments hostile to XMODEM.
X
X The XMODEM protocol extensions and YMODEM Batch address these
X weaknesses while maintaining XMODEM's simplicity.
X
X YMODEM is supported by the public domain programs YAM (CP/M),
X YAM(CP/M-86), YAM(CCPM-86), IMP (CP/M), KMD (CP/M), rz/sz (Unix, Xenix,
X VMS, Berkeley Unix, Venix, Xenix, Coherent, IDRIS, Regulus). Commerical
X implementations include MIRROR, and Professional-YAM.[1] Communications
X programs supporting these extensions have been in use since 1981.
X
X The 1k packet length (XMODEM-1k) described below may be used in
X conjunction with YMODEM Batch Protocol, or with single file transfers
X identical to the XMODEM/CRC protocol except for minimal
X changes to support 1k packets.

X
X	Another extension is simply called the g option. It provides maximum
X	throughput when used with end to end error correcting media,
X	such as X.PC and error correcting modems, including the emerging
X	9600 bps units by Electronic Vaults and others.
X
X	To complete this tome, Ward Christensen's original protocol document and
X	John Byrns's CRC-16 document are included for reference.
X
X	References to the MODEM or MODEM7 protocol have been changed
X	to XMODEM to accommodate the vernacular. In Australia, it is
X	properly called the Christensen Protocol.
X
X
X
X
X
X	----------
X
X	 1. Available for IBM PC,XT,AT, Unix and Xenix
X
X
X
X	Chapter 2
X
X
X
X
X
X
X
X	X/YMODEM Protocol Reference 09-11-86 4
X
X
X
X	2.1 Some Messages from the Pioneer
X
X	#: 130940 S0/Communications 25-Apr-85 18:38:47
X	Sb: my protocol
X	Fm: Ward Christensen 76703,302 (EDITED)
X	To: all
X
X	Be aware the article[2] DID quote me correctly in terms of the
X	phrases like "not robust", etc.
X
X	It was a quick hack I threw together, very unplanned (like
X	everything I do), to satisfy a personal need to communicate with
X	"some other" people.
X
X	ONLY the fact that it was done in 8/77, and that I put it in the public
X	domain immediately, made it become the standard that it is.

X
X	I think its time for me to
X
X	(1) document it; (people call me and say "my product is going
X	 to include it - what can I 'reference'", or "I'm writing a
X	 paper on it, what do I put in the bibliography") and
X
X	(2) propose an "incremental extension" to it, which might take
X	 "exactly" the form of Chuck Forsberg's YAM protocol. He
X	 wrote YAM in C for CP/M and put it in the public domain,
X	 and wrote a batch protocol for Unix[3] called rb and sb
X	 (receive batch, send batch), which was basically XMODEM with
X	 (a) a record 0 containing filename date time and size
X	 (b) a 1K block size option
X	 (c) CRC-16.
X
X	He did some clever programming to detect false ACK or EOT, but basically
X	left them the same.
X
X	People who suggest I make SIGNIFICANT changes to the protocol,
X	such as "full duplex", "multiple outstanding blocks", "multiple
X	destinations", etc etc don't understand that the incredible
X	simplicity of the protocol is one of the reasons it survived to
X	this day in as many machines and programs as it may be found in!
X
X	Consider the PC-NET group back in '77 or so - documenting to
X	beat the band
X	- THEY had a protocol, but it was "extremely complex", because
X	 it tried to be "all things to all people" - i.e. send binary
X	 files on a 7-bit system, etc. I was not that "benevolent". I
X	 (emphasize > I <) had an 8-bit UART,
X
X
X	----------
X
X	 2. Infoworld April 29 p. 16
X
X	 3. VAX/VMS versions of these programs are also available.
X
X
X
X	Chapter 2
X
X
X
X
X
X
X
X	X/YMODEM Protocol Reference	09-11-86	5
X
X

X
X	so "my protocol was an 8-bit protocol", and I would just say "sorry" to
X	people who were held back by 7-bit limitations. ...
X
X	Block size: Chuck Forsberg created an extension of my protocol,
X	called YAM, which is also supported via his public domain
X	programs for UNIX called rb and sb - receive batch and send
X	batch. They cleverly send a "block 0" which contains the
X	filename, date, time, and size. Unfortunately, its UNIX style,
X	and is abit weird[4] - octal numbers, etc. BUT, it is a nice way
X	to overcome the kludgy "echo the chars of the name" introduced
X	with MODEM7. Further, chuck uses CRC-16 and optional 1K blocks.
X	Thus the record 0, 1K, and CRC, make it a "pretty slick new
X	protocol" which is not significantly different from my own.
X
X	Also, there is a catchy name - YMODEM. That means to some
X	that it is the "next thing after XMODEM", and to others that it
X	is the Y(am)MODEM protocol. I don't want to emphasize that too
X	much - out of fear that other mfgrs might think it is a
X	"competitive" protocol, rather than an "unaffiliated" protocol.
X	Chuck is currently selling a much-enhanced version of his
X	CP/M-80 C program YAM, calling it Professional Yam, and its
X	for the PC - I'm using it right now. VERY slick! 32K capture buffer,
X	script, scrolling, previously captured text search, plus
X	built-in commands for just about everything - directory (sorted
X	every which way), XMODEM, YMODEM, KERMIT, and ASCII file
X	upload/download, etc. You can program it to "behave" with most
X	any system - for example when trying a number for CIS it detects
X	the "busy" string back from the modem and substitutes a diff
X	phone # into the dialing string and branches back to try it.
X
X
X
X	3. XMODEM PROTOCOL ENHANCEMENTS
X
X	This chapter discusses the protocol extensions to Ward Christensen's
X	1982 XMODEM protocol description document.
X
X	The original document recommends the user be asked whether to continue
X	trying or abort after 10 retries. Most programs no longer ask the
X	operator whether he wishes to keep retrying. Virtually all correctable
X	errors are corrected within the first few retransmissions. If
X	the line is so bad that ten attempts are insufficient, there is
X	a significant danger of undetected errors. If the connection is
X	that bad, it's better to redial for a better connection, or
X	mail a floppy disk.
X
X
X
X
X
X	----------
X

X   4. The file length, time, and file mode are optional.The pathname and
X   file length may be sent alone if desired.
X
X
X
X
X   Chapter   3   XMODEM Protocol Enhancements
X
X
X
X
X
X
X
X   X/YMODEM Protocol Reference   09-11-86   6
X
X
X
X   3.1 Graceful Abort
X
X   The YAM and Professional-YAM X/YMODEM routines recognize a
X   sequence of two consecutive CAN (Hex 18) characters without modem
X   errors (overrun, framing, etc.) as a transfer abort command. This
X   sequence is recognized when YAM is waiting for the beginning of a
X   packet or for an acknowledge to one that has been sent. The
X   check for two consecutive CAN characters virtually eliminates the
X   possibility of a line hit aborting the transfer. YAM sends
X   eight CAN characters when it aborts an XMODEM, YMODEM, or ZMODEM
X   protocol file transfer. Pro-YAM then sends eight backspaces to delete
X   the CAN characters from the remote's keyboard input buffer, in
X   case the remote had already aborted the transfer and was
X   awaiting a keyboarded command.
X
X
X   3.2 CRC-16 Option
X
X   The XMODEM protocol uses an optional two character CRC-16
X   instead of the one character arithmetic checksum used by the original
X   protocol and by most commercial implementations. CRC-16
X   guarantees detection of all single and double bit errors, all errors
X   with an odd number of error bits, all burst errors of length 16 or
X   less, 99.9969% of all 17-bit error bursts, and 99.9984 per cent
X   of all possible longer error bursts. By contrast, a double bit
X   error, or a burst error of 9 bits or more can sneak past the
X   XMODEM protocol arithmetic checksum.
X
X   The XMODEM/CRC protocol is similar to the XMODEM protocol,
X   except that the receiver specifies CRC-16 by sending C (Hex
X   43) instead of NAK when requesting the FIRST packet. A two
X   byte CRC is sent in place of the one byte arithmetic checksum.
X
X   YAM's c option to the r command enables CRC-16 in single file
X   reception, corresponding to the original implementation in the

X MODEM7 series programs. This remains the default because many
X commercial communications programs and bulletin board systems still do
X not support CRC-16, especially those written in Basic or Pascal.
X
X XMODEM protocol with CRC is accurate provided both sender and
X receiver both report a successful transmission. The protocol is robust
X in the presence of characters lost by buffer overloading on
X timesharing systems.
X
X The single character ACK/NAK responses generated by the
X receiving program adapt well to split speed modems, where the reverse
X channel is limited to ten per cent or less of he main channel's speed.
X
X XMODEM and YMODEM are half duplex protocols which do not attempt to
X transmit information and control signals in both directions at the same
X time. This avoids buffer overrun problems that have been reported by
X users attempting to exploit full duplex asynchronous file transfer
X protocols such as Blast.
X
X Professional-YAM adds several proprietary logic enhancements to XMODEM's
X
X
X
X Chapter 3 XMODEM Protocol Enhancements
X
X
X
X
X
X
X
X X/YMODEM Protocol Reference 09-11-86 7
X
X
X
X error detection and recovery. These compatible enhancements eliminate
X most of the bad file transfers other programs make when using
X the XMODEM protocol under less than ideal conditions.
X
X
X 3.3 XMODEM-1k 1024 Byte Packet
X
X The choice to use 1024 byte packets is expressed to the sending
X program on its command line or selection menu.[1] 1024 byte packets
X improve throughput in many applications, but some environments cannot
X accept 1024 byte bursts, especially minicomuters running 19.2kb
X ports.
X
X An STX (02) replaces the SOH (01) at the beginning of the transmitted
X block to notify the receiver of the longer packet length. The
X transmitted packet contains 1024 bytes of data. The receiver
X should be able to accept any mixture of 128 and 1024 byte
X packets. The packet number is incremented by one for each

X packet regardless of the packet length.
X
X The sender must not change between 128 and 1024 byte packet lengths if it
X has not received a valid ACK for the current packet. Failure to observe
X this restriction allows certain transmission errors to pass undetected.
X
X If 1024 byte packets are being used, it is possible for a file to "grow"
X up to the next multiple of 1024 bytes. This does not waste disk space if
X the allocation granularity is 1k or greater. With YMODEM batch
X transmission, the optional file length transmitted in the file
X name packet allows the receiver to discard the padding,
X preserving the exact file length and contents.
X
X 1024 byte packets may be used with batch file transmission or with single
X file transmission. CRC-16 should be used with the k option to preserve
X data integrity over phone lines. If a program wishes to enforce this
X recommendation, it should cancel the transfer, then issue an informative
X diagnostic message if the receiver requests checksum instead of CRC-16.
X Under no circumstances may a sending program use CRC-16 unless the
X receiver commands CRC-16.
X
X
X 4. YMODEM Batch File Transmission
X
X The YMODEM Batch protocol is an extension to the XMODEM/CRC protocol that
X allows 0 or more files to be transmitted with a single command. (Zero
X files may be sent if none of the requested files is accessible.) The
X design approach of the YMODEM Batch protocol is to use the
X normal routines for sending and receiving XMODEM packets in a
X layered fashion similar to
X
X
X ----------
X
X 1. See "KMD/IMP Exceptions to YMODEM" below.
X
X
X
X
X Chapter 4 XMODEM Protocol Enhancements
X
X
X
X
X
X
X
X X/YMODEM Protocol Reference 09-11-86 8
X
X
X
X Figure 1. 1024 byte Packets
X

```
X          SENDER                                    RECEIVER
X                                                    "s -k foo.bar"
X          "foo.bar open x.x minutes"
X                                                    C
X          STX 01 FE Data[1024] CRC CRC
X                                                    ACK
X          STX 02 FD Data[1024] CRC CRC
X                                                    ACK
X          STX 03 FC Data[1000] CPMEOF[24] CRC CRC
X                                                    ACK
X          EOT
X                                                    ACK
X
X         Figure 2.  Mixed 1024 and 128 byte Packets
X
X          SENDER                                    RECEIVER
X                                                    "s -k foo.bar"
X          "foo.bar open x.x minutes"
X                                                    C
X          STX 01 FE Data[1024] CRC CRC
X                                                    ACK
X          STX 02 FD Data[1024] CRC CRC
X                                                    ACK
X          SOH 03 FC Data[128] CRC CRC
X                                                    ACK
X          SOH 04 FB Data[100] CPMEOF[28] CRC CRC
X                                                    ACK
X          EOT
X                                                    ACK
X
X      packet switching methods.
X
X      Why was it necessary to design a new batch protocol when one already
X      existed in MODEM7?[1] The batch file mode used by MODEM7 is unsuitable
X      because it does not permit full pathnames, file length, file date, or
X      other attribute information to be transmitted. Such a restrictive design,
X      hastily implemented with only CP/M in mind, would not have permitted
X      extensions to current areas of personal computing such as Unix, DOS, and
X      object oriented systems.  In addition, the MODEM7 batch file mode is
X      somewhat susceptible to transmission impairments.
X
X
X
X      ----------
X
X       1. The MODEM7 batch protocol transmitted CP/M FCB bytes f1...f8 and
X          t1...t3 one character at a time.  The receiver echoed these bytes as
X          received, one at a time.
X
X
X
X
X      Chapter  4                              XMODEM Protocol Enhancements
```

As in the case of single a file transfer, the receiver initiates batch file transmission by sending a "C" character (for CRC-16).

The sender opens the first file and sends packet number 0 with the following information.[2]

Only the pathname (file name) part is required for batch transfers.

To maintain upwards compatibility, all unused bytes in packet 0 must be set to null.

Pathname The pathname (conventionally, the file name) is sent as a null terminated ASCII string. This is the filename format used by the handle oriented MSDOS(TM) functions and C library fopen functions. An assembly language example follows:

```
DB      'foo.bar',0
```

No spaces are included in the pathname. Normally only the file name stem (no directory prefix) is transmitted unless the sender has selected YAM's f option to send the full pathname. The source drive (A:, B:, etc.) is never sent.

Filename Considerations:

+ File names should be translated to lower case unless the sending system supports upper/lower case file names. This is a convenience for users of systems (such as Unix) which store filenames in upper and lower case.

+ The receiver should accommodate file names in lower and upper case.

+ When transmitting files between different operating systems, file names must be acceptable to both the sender and receiving operating systems.

If directories are included, they are delimited by /; i.e., "subdir/foo" is acceptable, "subdir\foo" is not.

Length The file length and each of the succeeding fields are optional.[3] The length field is stored in the packet as a decimal string counting the number of data bytes in the file. The file length does not include any CPMEOF (^Z) or other

X garbage characters used to pad the last packet.
X
X
X ----------
X
X 2. Only the data part of the packet is described here.
X
X 3. Fields may not be skipped.
X
X
X
X
X Chapter 4 XMODEM Protocol Enhancements
X
X
X
X
X
X
X
X X/YMODEM Protocol Reference 09-11-86 10
X
X
X
X If the file being transmitted is growing during transmission, the
X length field should be set to at least the final expected file
X length, or not sent.
X
X The receiver stores the specified number of characters, discarding
X any padding added by the sender to fill up the last packet.
X
X Modification Date The mod date is optional, and the filename and length
X may be sent without requiring the mod date to be sent.
X
X Iff the modification date is sent, a single space separates the
X modification date from the file length.
X
X The mod date is sent as an octal number giving the time the contents
X of the file were last changed, measured in seconds from Jan 1 1970
X Universal Coordinated Time (GMT). A date of 0 implies the
X modification date is unknown and should be left as the date the file
X is received.
X
X This standard format was chosen to eliminate ambiguities
X arising from transfers between different time zones.
X
X Two Microsoft blunders complicate the use of modification dates in
X file transfers with MSDOS(TM) systems. The first is the lack of
X timezone standardization in MS-DOS. A file's creation time can not
X be known unless the timezone of the system that wrote the file[4] is
X known. Unix solved this problem (for planet Earth, anyway) by
X stamping files with Universal Time (GMT). Microsoft would have to
X include the timezone of origin in the directory entries, but does

X not. Professional-YAM gets around this problem by using the z
X parameter which is set to the number of minutes local time lags GMT.
X For files known to originate from a different timezone, the -zT
X option may be used to specify T as the timezone for an individual
X file transfer.
X
X The second problem is the lack of a separate file creation date in
X DOS. Since some backup schemes used with DOS rely on the file
X creation date to select files to be copied to the archive, back-
X dating the file modification date could interfere with the safety of
X the transferred files. For this reason, Pro-YAM does not modify the
X date of received files with the header information unless the d
X numeric parameter is non zero.
X
X
X
X
X
X ----------
X
X 4. Not necessarily that of the transmitting system!
X
X
X
X
X Chapter 4 XMODEM Protocol Enhancements
X
X
X
X
X
X
X
X X/YMODEM Protocol Reference 09-11-86 11
X
X
X
X Mode Iff the file mode is sent, a single space separates the file mode
X from the modification date. The file mode is stored as an octal
X string. Unless the file originated from a Unix system, the
X file mode is set to 0. rb(1) checks the file mode for the
X 0x8000 bit which indicates a Unix type regular file. Files
X with the 0x8000 bit set are assumed to have been sent from
X another Unix (or similar) system which uses the same file
X conventions. Such files are not translated in any way.
X
X
X Serial Number Iff the serial number is sent, a single space separates the
X serial number from the file mode. The serial number of the
X transmitting program is stored as an octal string.
X Programs which do not have a serial number should omit this
X field, or set it to 0. The receiver's use of this field is
X optional.

X
X
X Other Fields YMODEM was designed to allow additional header fields to be
X added as above without creating compatibility problems with older
X YMODEM programs. Please contact Omen Technology if other fields are
X needed for special application requirements.
X
X The rest of the packet is set to nulls. This is essential to preserve
X upward compatibility.[5]
X
X If the filename packet is received with a CRC or other error, a
X restrnsmission is requested. After the filename packet has been
X received, it is ACK'ed if the write open is successful. If the
X file cannot be opened for writing, the receiver cancels the
X transfer with CAN characters as described above.
X
X The receiver then initiates transfer of the file contents
X according to the standard XMODEM/CRC protocol.
X
X After the file contents have been transmitted, the receiver
X again asks for the next pathname. Transmission of a null
X pathname terminates batch file transmission. Note that
X transmission of no files is not necessarily an error. This is
X possible if none of the files requested of the sender could be
X opened for reading.
X
X In batch transmission, the receiver automatically requests CRC-16.
X
X The Unix programs sz(1) and rz(1) included in the source code file
X
X
X ----------
X
X 5. If, perchance, this information extends beyond 128 bytes (possible
X with Unix 4.2 BSD extended file names), the packet should be
X sent as a 1k packet as described above.
X
X
X
X
X Chapter 4 XMODEM Protocol Enhancements
X
X
X
X
X
X
X
X
X X/YMODEM Protocol Reference 09-11-86 12
X
X
X
X RZSZ[12].SHQ (rzsz1.sh, rzsz2.sh) should answer other questions about

X   YMODEM batch protocol.
X
X   Figure 3. YMODEM Batch Transmission Session
X
X   SENDER   RECEIVER
X   "sb foo.*<CR>"
X   "sending in batch mode etc."
X   C (command:rb)
X   SOH 00 FF foo.c NUL[123] CRC CRC
X   ACK
X   C
X   SOH 01 FE Data[128] CRC CRC
X   ACK
X   STX 02 FD Data[1024] CRC CRC
X   ACK
X   SOH 03 FC Data[128] CRC CRC
X   ACK
X   SOH 04 FB Data[100] CPMEOF[28] CRC CRC
X   ACK
X   EOT
X   NAK
X   EOT
X   ACK
X   C
X   SOH 00 FF NUL[128] CRC CRC
X   ACK
X
X   Figure 4. YMODEM Filename packet transmitted by sz
X
X   -rw-r--r-- 6347 Jun 17 1984 20:34 bbcsched.txt
X
X   00 0100FF62 62637363 6865642E 74787400 |...bbcsched.txt.|
X   10 36333437 20333331 34373432 35313320 |6347 3314742513 |
X   20 31303036 34340000 00000000 00000000 |100644..........|
X   30 00000000 00000000 00000000 00000000
X   80 000000CA 56
X
X
X

X
X
X
X Figure 5. YMODEM Header Information used by various programs
X

Program	Batch	Length	Date	Mode	S/N	1k-Blk	g-Option
Unix rz/sz	yes	yes	yes	yes	no	yes	sb only
VMS rb/sb	yes	yes	no	no	no	yes	no
Pro-YAM	yes	yes	yes	no	yes	yes	yes
CP/M YAM	yes	no	no	no	no	yes	no
KMD/IMP	yes	?	no	no	no	yes	no

X
X 4.1 KMD/IMP Exceptions to YMODEM
X
X KMD and IMP character sequence emitted by the receiver (CK) to
X automatically trigger the use of 1024 byte packets as an alternative to
X specifying this option on this command line. Although this two character
X sequence works well on single process micros in direct communication,
X timesharing systems and packet switched networks can separate the
X successive characters by several seconds, rendering this method
X unreliable.
X
X Sending programs may detect the CK sequence if the operating enviornment
X does not preclude reliable implementation.
X
X Receiving programs should retain the option of sending the
X standard YMODEM C (not CK) trigger with the standard 10 second
X timeout to insure compatibility with newer forms of
X telecommunications technology.
X
X
X
X 5. YMODEM-g File Transmission
X
X Developing technology is providing phone line data transmission at ever
X higher speeds using very specialized techniques. These high
X speed modems, as well as session protocols such as X.PC, provide

X high speed, error free communications at the expense of
X considerably increased delay time.
X
X This delay time is moderate compared to human interactions, but it
X cripples the throughput of most error correcting protocols.
X
X The g option to YMODEM has proven effective under these circumstances.
X The g option is driven by the receiver, which initiates the
X batch transfer by transmitting a G instead of C. When the
X sender recognizes the G, it bypasses the usual wait for an ACK
X to each transmitted packet, sending succeeding packets at full
X speed, subject to XOFF/XON or other flow control exerted by the medium.
X
X
X
X Chapter 5 XMODEM Protocol Enhancements
X
X
X
X
X
X
X
X X/YMODEM Protocol Reference 09-11-86 14
X
X
X
X The sender expects an inital G to initiate the transmission of a
X particular file, and also expects an ACK on the EOT sent at the end of
X each file. This synchronization allows the receiver time to open and
X close files as necessary.
X
X If an error is detected in a YMODEM-g transfer, the receiver aborts the
X transfer with the multiple CAN abort sequence. The ZMODEM protocol should
X be used in applications that require both streaming throughput and error
X recovery.
X
X Figure 6. YMODEM-g Transmission Session
X
X SENDER RECEIVER
X "sb foo.*<CR>"
X "sending in batch mode etc..."
X G (command:rb -g)
X SOH 00 FF foo.c NUL[123] CRC CRC
X G
X SOH 01 FE Data[128] CRC CRC
X STX 02 FD Data[1024] CRC CRC
X SOH 03 FC Data[128] CRC CRC
X SOH 04 FB Data[100] CPMEOF[28] CRC CRC
X EOT
X ACK
X G
X SOH 00 FF NUL[128] CRC CRC

X
X
X 6. XMODEM PROTOCOL OVERVIEW
X
X 8/9/82 by Ward Christensen.
X
X I will maintain a master copy of this. Please pass on changes or
X suggestions via CBBS/Chicago at (312) 545-8086, CBBS/CPMUG (312) 849-1132
X or by voice at (312) 849-6279.
X
X 6.1 Definitions
X
X <soh> 01H
X <eot> 04H
X <ack> 06H
X <nak> 15H
X <can> 18H
X <C> 43H
X
X
X
X
X
X
X
X
X
X
X
X Chapter 6 Xmodem Protocol Overview
X
X
X
X
X
X
X
X X/YMODEM Protocol Reference 09-11-86 15
X
X
X
X 6.2 Transmission Medium Level Protocol
X
X Asynchronous, 8 data bits, no parity, one stop bit.
X
X The protocol imposes no restrictions on the contents of the data being
X transmitted. No control characters are looked for in the 128-byte data
X messages. Absolutely any kind of data may be sent - binary, ASCII, etc.
X The protocol has not formally been adopted to a 7-bit environment for the
X transmission of ASCII-only (or unpacked-hex) data , although it could be
X simply by having both ends agree to AND the protocol-dependent data with
X 7F hex before validating it. I specifically am referring to the
X checksum, and the block numbers and their ones- complement.
X

```
X      Those wishing to maintain compatibility of the CP/M file structure, i.e.
X      to allow modemming ASCII files to or from CP/M systems should follow this
X      data format:
X
X         + ASCII tabs used (09H); tabs set every 8.
X
X         + Lines terminated by CR/LF (0DH 0AH)
X
X         + End-of-file indicated by ^Z, 1AH. (one or more)
X
X         + Data is variable length, i.e. should be considered a continuous
X           stream of data bytes, broken into 128-byte chunks purely for the
X           purpose of transmission.
X
X         + A CP/M "peculiarity": If the data ends exactly on a 128-byte
X           boundary, i.e. CR in 127, and LF in 128, a subsequent sector
X           containing the ^Z EOF character(s) is optional, but is preferred.
X           Some utilities or user programs still do not handle EOF without ^Zs.
SHAR_EOF
echo "End of part 6"
echo "File YMODEM.DOC is continued in part 7"
echo "7" > s2_seq_.tmp
exit 0


From 9061157546281148518
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,64d7ac6b1c1c040e,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-30 09:16:11 PST
Path: utzoo!attcan!uunet!husc6!mailrus!rutgers!ucsd!ucbvax!agate!saturn!ssyx!
koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i047: zmdm -- terminal program with file transfer  part07/08
Keywords: mwc, alcyon, shar
Message-ID: <3513@saturn.ucsc.edu>
Date: 30 May 88 16:16:11 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1358
Approved: koreth@ssyx.ucsc.edu

Submitted-by: bammi@mandrill.ces.cwru.edu (Jwahar R. Bammi)
Posting-number: Volume 1, Issue 47
Archive-name: zmdm/part07

#!/bin/sh
# this is part 7 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file YMODEM.DOC continued
#
CurArch=7
if test ! -r s2_seq_.tmp
```

```
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file YMODEM.DOC"
sed 's/^X//' << 'SHAR_EOF' >> YMODEM.DOC
X
X         + The last block sent is no different from others, i.e.  there is no
X           "short block".
X              Figure 7.  XMODEM Message Block Level Protocol
X
X      Each block of the transfer looks like:
X            <SOH><blk #><255-blk #><--128 data bytes--><cksum>
X      in which:
X      <SOH>           = 01 hex
X      <blk #>         = binary number, starts at 01 increments by 1, and
X                wraps 0FFH to 00H (not to 01)
X      <255-blk #>   =   blk # after going thru 8080 "CMA" instr, i.e.
X                each bit complemented in the 8-bit block number.
X                Formally, this is the "ones complement".
X      <cksum>       = the sum of the data bytes only.  Toss any carry.
X
X
X
X
X
X
X
X      Chapter   6                                  Xmodem Protocol Overview
X
X
X
X
X
X
X
X      X/YMODEM Protocol Reference        09-11-86                        16
X
X
X
X      6.3  File Level Protocol
X
X      6.3.1  Common_to_Both_Sender_and_Receiver
X      All errors are retried 10 times.  For versions running with an operator
X      (i.e. NOT with XMODEM), a message is typed after 10 errors asking the
X      operator whether to "retry or quit".
X
X      Some versions of the protocol use <can>, ASCII ^X, to cancel
```

X transmission. This was never adopted as a standard, as having a
X single "abort" character makes the transmission susceptible to
X false termination due to an <ack> <nak> or <soh> being corrupted
X into a <can> and aborting transmission.
X
X The protocol may be considered "receiver driven", that is, the
X sender need not automatically re-transmit, although it does in
X the current implementations.
X
X
X 6.3.2 Receive_Program_Considerations
X The receiver has a 10-second timeout. It sends a <nak> every time it
X times out. The receiver's first timeout, which sends a <nak>,
X signals the transmitter to start. Optionally, the receiver
X could send a <nak> immediately, in case the sender was ready.
X This would save the initial 10 second timeout. However, the
X receiver MUST continue to timeout every 10 seconds in case the
X sender wasn't ready.
X
X Once into a receiving a block, the receiver goes into a
X one-second timeout for each character and the checksum. If the
X receiver wishes to <nak> a block for any reason (invalid header,
X timeout receiving data), it must wait for the line to clear.
X See "programming tips" for ideas.
X
X Synchronizing: If a valid block number is received, it will be: 1) the
X expected one, in which case everything is fine; or 2) a repeat of the
X previously received block. This should be considered OK, and only
X indicates that the receivers <ack> got glitched, and the sender re-
X transmitted; 3) any other block number indicates a fatal loss of
X synchronization, such as the rare case of the sender getting a
X line-glitch that looked like an <ack>. Abort the transmission,
X sending a <can>
X
X
X 6.3.3 Sending_program_considerations
X While waiting for transmission to begin, the sender has only a
X single very long timeout, say one minute. In the current
X protocol, the sender has a 10 second timeout before retrying. I
X suggest NOT doing this, and letting the protocol be completely
X receiver-driven. This will be compatible with existing programs.
X
X When the sender has no more data, it sends an <eot>, and awaits an <ack>,
X resending the <eot> if it doesn't get one. Again, the protocol could be
X receiver-driven, with the sender only having the high-level 1-minute
X timeout to abort.
X
X
X
X

Here is a sample of the data flow, sending a 3-block message.
It includes the two most common line hits - a garbaged block,
and an <ack> reply getting garbaged. <xx> represents the checksum byte.

Figure 8. Data flow including Error Recovery

```
SENDER                          RECEIVER
                        times out after 10 seconds,
                        <---            <nak>
<soh> 01 FE -data- <xx>      --->
                        <---            <ack>
<soh> 02 FD -data- xx        --->        (data gets line hit)
                        <---            <nak>
<soh> 02 FD -data- xx        --->
                        <---            <ack>
<soh> 03 FC -data- xx        --->
(ack gets garbaged)          <---            <ack>
<soh> 03 FC -data- xx        --->            <ack>
<eot>                        --->
                        <---       <anything except ack>
<eot>                        --->
                        <---            <ack>
(finished)
```

6.4 Programming Tips

+ The character-receive subroutine should be called with a parameter specifying the number of seconds to wait. The receiver should first call it with a time of 10, then <nak> and try again, 10 times.

  After receiving the <soh>, the receiver should call the character receive subroutine with a 1-second timeout, for the remainder of the message and the <cksum>. Since they are sent as a continuous stream, timing out of this implies a serious like glitch that caused, say, 127 characters to be seen instead of 128.

+ When the receiver wishes to <nak>, it should call a "PURGE" subroutine, to wait for the line to clear. Recall the sender tosses any characters in its UART buffer immediately upon completing sending a block, to ensure no glitches were mis-interpreted.

  The most common technique is for "PURGE" to call the character

X receive subroutine, specifying a 1-second timeout,[1] and looping
X back to PURGE until a timeout occurs. The <nak> is then sent,
X ensuring the other end will see it.
X
X
X ----------
X
X 1. These times should be adjusted for use with timesharing systems.
X
X
X
X
X Chapter 6 Xmodem Protocol Overview
X
X
X
X
X
X
X
X X/YMODEM Protocol Reference 09-11-86 18
X
X
X
X + You may wish to add code recommended by John Mahr to your character
X receive routine - to set an error flag if the UART shows framing
X error, or overrun. This will help catch a few more glitches - the
X most common of which is a hit in the high bits of the byte in two
X consecutive bytes. The <cksum> comes out OK since counting
X in 1-byte produces the same result of adding 80H + 80H as
X with adding 00H + 00H.
X
X
X
X 7. XMODEM/CRC Overview
X
X 1/13/85 by John Byrns -- CRC option.
X
X Please pass on any reports of errors in this document or suggestions for
X improvement to me via Ward's/CBBS at (312) 849-1132, or by voice at (312)
X 885-1105.
X
X The CRC used in the Modem Protocol is an alternate form of block check
X which provides more robust error detection than the original checksum.
X Andrew S. Tanenbaum says in his book, Computer Networks, that the CRC-
X CCITT used by the Modem Protocol will detect all single and double bit
X errors, all errors with an odd number of bits, all burst errors of length
X 16 or less, 99.997% of 17-bit error bursts, and 99.998% of 18-bit and
X longer bursts.
X
X The changes to the Modem Protocol to replace the checksum with
X the CRC are straight forward. If that were all that we did we
X would not be able to communicate between a program using the old

X checksum protocol and one using the new CRC protocol. An initial
X handshake was added to solve this problem. The handshake allows
X a receiving program with CRC capability to determine whether the
X sending program supports the CRC option, and to switch it to CRC
X mode if it does. This handshake is designed so that it will work
X properly with programs which implement only the original
X protocol. A description of this handshake is presented in section 10.
X
X Figure 9. Message Block Level Protocol, CRC mode
X
X Each block of the transfer in CRC mode looks like:
X <SOH><blk #><255-blk #><--128 data bytes--><CRC hi><CRC lo>
X in which:
X <SOH> = 01 hex
X <blk #> = binary number, starts at 01 increments by 1, and
X wraps 0FFH to 00H (not to 01)
X <255-blk #> = ones complement of blk #.
X <CRC hi> = byte containing the 8 hi order coefficients of the CRC.
X <CRC lo> = byte containing the 8 lo order coefficients of the CRC.
X
X
X
X
X
X
X
X Chapter 7 Xmodem Protocol Overview
X
X
X
X
X
X
X
X X/YMODEM Protocol Reference 09-11-86 19
X
X
X
X 7.1 CRC Calculation
X
X 7.1.1 Formal_Definition
X To calculate the 16 bit CRC the message bits are considered to be the
X coefficients of a polynomial. This message polynomial is first multiplied
X by X^16 and then divided by the generator polynomial (X^16 + X^12 + X^5 +
X 1) using modulo two arithmetic. The remainder left after the division is
X the desired CRC. Since a message block in the Modem Protocol is 128 bytes
X or 1024 bits, the message polynomial will be of order X^1023.
X The hi order bit of the first byte of the message block is the
X coefficient of X^1023 in the message polynomial. The lo order
X bit of the last byte of the message block is the coefficient of
X X^0 in the message polynomial.
X
X Figure 10. Example of CRC Calculation written in C

X UPDCRC update routine from "rbsb.c". Refer to the source code for these
X programs (contained in RZSZ1.SHQ and RZSZ2.SHQ) for usage. A fast table
X driven macro is also included in this file.

```
X /* update CRC */
X unsigned short
X updcrc(c, crc)
X register c;
X register unsigned crc;
X {
X   register count;
X
X   for (count=8; --count>=0;) {
X           if (crc & 0x8000) {
X                   crc <<= 1;
X                   crc += (((c<<=1) & 0400)  !=  0);
X                   crc ^= 0x1021;
X           }
X           else {
X                   crc <<= 1;
X                   crc += (((c<<=1) & 0400)  !=  0);
X           }
X   }
X   return crc;
X }
```

X
X 7.2 CRC File Level Protocol Changes
X
X 7.2.1 Common_to_Both_Sender_and_Receiver
X The only change to the File Level Protocol for the CRC option is the
X initial handshake which is used to determine if both the sending and the
X receiving programs support the CRC mode. All Modem Programs
X should support the checksum mode for compatibility with older
X versions. A receiving program that wishes to receive in CRC
X mode implements the mode setting handshake by sending a <C> in
X place of the initial <nak>. If the sending program supports CRC
X mode it will recognize the <C> and will set itself into CRC
X mode, and respond by sending the first block as if a <nak> had
X been received. If the sending program does not support CRC mode it will
X
X
X

X not respond to the <C> at all. After the receiver has sent the
X <C> it will wait up to 3 seconds for the <soh> that starts the
X first block. If it receives a <soh> within 3 seconds it will
X assume the sender supports CRC mode and will proceed with the
X file exchange in CRC mode. If no <soh> is received within 3
X seconds the receiver will switch to checksum mode, send a <nak>,
X and proceed in checksum mode. If the receiver wishes to use
X checksum mode it should send an initial <nak> and the sending program
X should respond to the <nak> as defined in the original Modem Protocol.
X After the mode has been set by the initial <C> or <nak> the protocol
X follows the original Modem Protocol and is identical whether the checksum
X or CRC is being used.
X
X
X 7.2.2 Receive_Program_Considerations
X There are at least 4 things that can go wrong with the mode setting
X handshake.
X
X 1. the initial <C> can be garbled or lost.
X
X 2. the initial <soh> can be garbled.
X
X 3. the initial <C> can be changed to a <nak>.
X
X 4. the initial <nak> from a receiver which wants to receive
X in checksum can be changed to a <C>.
X
X The first problem can be solved if the receiver sends a second <C> after
X it times out the first time. This process can be repeated several times.
X It must not be repeated too many times before sending a <nak> and
X switching to checksum mode or a sending program without CRC support may
X time out and abort. Repeating the <C> will also fix the second problem if
X the sending program cooperates by responding as if a <nak> were received
X instead of ignoring the extra <C>.
X
X It is possible to fix problems 3 and 4 but probably not worth the trouble
X since they will occur very infrequently. They could be fixed by switching
X modes in either the sending or the receiving program after a large number
X of successive <nak>s. This solution would risk other problems however.
X
X
X 7.2.3 Sending_Program_Considerations
X The sending program should start in the checksum mode. This will insure
X compatibility with checksum only receiving programs. Anytime a <C> is
X received before the first <nak> or <ack> the sending program should set
X itself into CRC mode and respond as if a <nak> were received. The sender
X should respond to additional <C>s as if they were <nak>s until the first
X <ack> is received. This will assist the receiving program in determining
X the correct mode when the <soh> is lost or garbled. After the first <ack>
X is received the sending program should ignore <C>s.
X
X
X

X
X
X
X 7.3 Data Flow Examples with CRC Option
X
X Here is a data flow example for the case where the receiver requests
X transmission in the CRC mode but the sender does not support the CRC
X option. This example also includes various transmission errors. <xx>
X represents the checksum byte.
X
X Figure 11. Data Flow: Receiver has CRC Option, Sender Doesn't
X
X SENDER RECEIVER
X <--- <C>
X times out after 3 seconds,
X <--- <C>
X times out after 3 seconds,
X <--- <C>
X times out after 3 seconds,
X <--- <C>
X times out after 3 seconds,
X <--- <nak>
X <soh> 01 FE -data- <xx> --->
X <--- <ack>
X <soh> 02 FD -data- <xx> ---> (data gets line hit)
X <--- <nak>
X <soh> 02 FD -data- <xx> --->
X <--- <ack>
X <soh> 03 FC -data- <xx> --->
X (ack gets garbaged) <--- <ack>
X times out after 10 seconds,
X <--- <nak>
X <soh> 03 FC -data- <xx> --->
X <--- <ack>
X <eot> --->
X <--- <ack>
X
X Here is a data flow example for the case where the receiver requests
X transmission in the CRC mode and the sender supports the CRC
X option. This example also includes various transmission errors.
X <xxxx> represents the 2 CRC bytes.
X

X
X
X

X A version this file with boldface, underlining, and superscripts for
X printing on Epson or Gemini printers is available on Telegodzilla as
X "YMODEME.DOC" or "YMODEME.DQC".
X
X UUCP sites can obtain this file with
X uucp omen!/usr/spool/uucppublic/ymodem.doc /tmp
X A continually updated list of available files is stored in
X /usr/spool/uucppublic/FILES.
X
X The following L.sys line calls Telegodzilla (Pro-YAM in host operation).
X Telegodzilla waits for carriage returns to determine the incoming speed.
X If none is detected, 1200 bps is assumed and a greeting is displayed.
X
X In response to "Name Please:" uucico gives the Pro-YAM "link"
X command as a user name. The password (Giznoid) controls access
X to the Xenix system connected to the IBM PC's other serial port.
X Communications between Pro-YAM and Xenix use 9600 bps; YAM
X converts this to the caller's speed.
X
X Finally, the calling uucico logs in as uucp.
X
X omen Any ACU 1200 1-503-621-3746 se:--se: link ord: Giznoid in:--in: uucp
X
X Contact omen!caf if you wish the troff source files.
X
X
X
X
X
X Chapter 8 Xmodem Protocol Overview
X
X
X
X
X
X
X
X X/YMODEM Protocol Reference 09-11-86 23
X
X
X
X 9. Document Revisions
X
X The September 11 1986 edition clarifies nomenclature and some minor
X points. The April 15 1986 edition clarifies some points concerning CRC
X calculations and spaces in the header.
X
X
X 10. YMODEM Programs
X
X A demonstration version of Professional-YAM is available as ZMODEM.ARC
X This may be used to test YMODEM amd ZMODEM implementations.
X

X Unix programs supporting the YMODEM protocol are available on
X Telegodzilla as RZSZ1.SHQ and RZSZ2.SHQ (SQueezed shell archives).
X Most Unix like systems are supported, including V7, Xenix, Sys III,
X 4.2 BSD, SYS V, Idris, Coherent, and Regulus.
X
X A version for VAX-VMS is available in VRBSB.SHQ.
X
X Irv Hoff has added YMODEM 1k packets and basic YMODEM batch transfers to
X the KMD and IMP series programs, which replace the XMODEM and
X MODEM7/MDM7xx series respectively. Overlays are available for a wide
X variety of CP/M systems.
X
X Questions about YMODEM, the Professional-YAM communications program, and
X requests for evaluation copies may be directed to:
X Chuck Forsberg
X Omen Technology Inc
X 17505-V Sauvie Island Road
X Portland Oregon 97231
X Voice: 503-621-3406
X Modem: 503-621-3746 Speed: 2400,1200,300
X Usenet: ...!tektronix!reed!omen!caf
X Compuserve: 70007,2304
X Source: TCE022
X
X
X

X
X CONTENTS
X
X
X 1. ROSETTA STONE..... 2
X
X 2. YET ANOTHER PROTOCOL?..... 2
X 2.1 Some Messages from the Pioneer..... 4
X
X 3. XMODEM PROTOCOL ENHANCEMENTS..... 5
X 3.1 Graceful Abort..... 6
X 3.2 CRC-16 Option..... 6
X 3.3 XMODEM-1k 1024 Byte Packet..... 7
X
X 4. YMODEM Batch File Transmission..... 7
X 4.1 KMD/IMP Exceptions to YMODEM..... 13
X
X 5. YMODEM-g File Transmission..... 13
X
X 6. XMODEM PROTOCOL OVERVIEW..... 14
X 6.1 Definitions..... 14
X 6.2 Transmission Medium Level Protocol..... 15
X 6.3 File Level Protocol..... 16
X 6.4 Programming Tips..... 17
X
X 7. XMODEM/CRC Overview..... 18
X 7.1 CRC Calculation..... 19
X 7.2 CRC File Level Protocol Changes..... 19
X 7.3 Data Flow Examples with CRC Option..... 21
X
X 8. MORE INFORMATION..... 22
X
X 9. Document Revisions..... 23
X
X 10. YMODEM Programs..... 23
X
X
X
X
X
X
X
X
X
X
X
X
X
X
X
X
X
X
X
X
X
X

LIST OF FIGURES

Figure 1. 1024 byte Packets..... 7

Figure 2. Mixed 1024 and 128 byte Packets..... 7

Figure 3. YMODEM Batch Transmission Session..... 12

Figure 4. YMODEM Filename packet transmitted by sz..... 12

Figure 5. YMODEM Header Information used by various programs..... 13

Figure 6. YMODEM-g Transmission Session..... 14

Figure 7. XMODEM Message Block Level Protocol..... 15

Figure 8. Data flow including Error Recovery..... 17

Figure 9. Message Block Level Protocol, CRC mode..... 18

Figure 10. Example of CRC Calculation written in C..... 19

Figure 11. Data Flow: Receiver has CRC Option, Sender Doesn't..... 21

Figure 12. Receiver and Sender Both have CRC Option..... 22

```
X
X
X
X
X
X
X
X
X
X
X
X
X
X
X                                   - ii -
X
X
X
X
SHAR_EOF
echo "File YMODEM.DOC is complete"
chmod 0600 YMODEM.DOC || echo "restore of YMODEM.DOC fails"
echo "x - extracting ZM.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > ZM.C &&
X/*
X *               ACKNOWLEDGEMENTS
X *
X *  ZMDM was derived from rz/sz for Unix  posted by
X *  Chuck Forsberg (...!tektronix!reed!omen!caf ). We
X *  thank him for his excellent code, and for giving
X *  us permission to use and distribute his code and
X *  documentation.
X *
X *  Atari St version by:
X *       Jwahar Bammi
X *            usenet: mandrill!bammi@{decvax,sun}.UUCP
X *            csnet:  bammi@mandrill.ces.CWRU.edu
X *            arpa:   bammi@mandrill.ces.CWRU.edu
X *            CompuServe: 71515,155
X */
X
X/*
X *   Z M . C
X *    ZMODEM protocol primitives
X *    01-15-87  Chuck Forsberg Omen Technology Inc
X *
X * Entry point Functions:
X *  zsbhdr(type, hdr) send binary header
X *  zshhdr(type, hdr) send hex header
X *  zgethdr(hdr, eflag) receive header - binary or hex
X *  zsdata(buf, len, frameend) send data
X *  zrdata(buf, len) receive data
```

```
X *  stohdr(pos) store position data in Txhdr
X *  long rclhdr(hdr) recover position offset from header
X */
X#include "config.h"
X#include "zmdm.h"
X#include "common.h"
X
X#ifndef CANFDX
X#include "zmodem.h"
X#endif
X
X#define xsendline(X) sendline(X)
X
Xstatic char *frametypes[] = {
X    "Carrier Lost",         /* -3 */
X    "TIMEOUT",          /* -2 */
X    "ERROR",       /* -1 */
X#define FTOFFSET 3
X    "ZRQINIT",
X    "ZRINIT",
X    "ZSINIT",
X    "ZACK",
X    "ZFILE",
X    "ZSKIP",
X    "ZNAK",
X    "ZABORT",
X    "ZFIN",
X    "ZRPOS",
X    "ZDATA",
X    "ZEOF",
X    "ZFERR",
X    "ZCRC",
X    "ZCHALLENGE",
X    "ZCOMPL",
X    "ZCAN",
X    "ZFREECNT",
X    "ZCOMMAND",
X    "ZSTDERR",
X    "xxxxx"
X#define FRTYPES 22    /* Total number of frame types in this array */
X             /*  not including psuedo negative entries */
X};
X
X/* Send ZMODEM binary header hdr of type type */
Xzsbhdr(type, hdr)
Xregister int type;
Xregister char *hdr;
X{
X    register int n;
X    register unsigned int crc;
X
X    vfile2("zsbhdr: %s %lx", frametypes[type+FTOFFSET], rclhdr(hdr));
X    if (type == ZDATA)
```

```
X         for (n = Znulls; --n >=0; )
X              zsendline(0);
X
X    xsendline(ZPAD); xsendline(ZDLE);
X
X    if (Txfcs32)
X         zsbh32(hdr, type);
X    else {
X         xsendline(ZBIN); zsendline(type); crc = updcrc(type, 0);
X
X         for (n=4; --n >= 0;) {
X              zsendline(*hdr);
X              crc = updcrc((0377& *hdr++), crc);
X         }
X         crc = updcrc(0,updcrc(0,crc));
X         zsendline(crc>>8);
X         zsendline(crc);
X    }
X    if (type != ZDATA)
X         flush_modem();
X}
X
X
X/* Send ZMODEM binary header hdr of type type */
Xzsbh32(hdr, type)
Xregister char *hdr;
Xregister int type;
X{
X    register int n;
X    register unsigned long crc;
X
X    xsendline(ZBIN32);  zsendline(type);
X    crc = 0xFFFFFFFFL; crc = UPDC32(type, crc);
X
X    for (n=4; --n >= 0;) {
X         zsendline(*hdr);
X         crc = UPDC32((0377& *hdr++), crc);
X    }
X    crc = ~crc;
X    for (n=4; --n >= 0;) {
X         zsendline((unsigned int)(crc&0xFFFF));  crc >>= 8;
X    }
X}
X
X/* Send ZMODEM HEX header hdr of type type */
Xzshhdr(type, hdr)
Xregister int type;
Xregister char *hdr;
X{
X    register int n;
X    register unsigned int crc;
X
X    vfile2("zshhdr: %s %lx", frametypes[type+FTOFFSET], rclhdr(hdr));
```

```
X    sendline(ZPAD); sendline(ZPAD); sendline(ZDLE); sendline(ZHEX);
X    zputhex(type);
X
X    crc = updcrc(type, 0);
X    for (n=4; --n >= 0;) {
X         zputhex(*hdr); crc = updcrc((0377& *hdr++), crc);
X    }
X    crc = updcrc(0,updcrc(0,crc));
X    zputhex(crc>>8); zputhex(crc);
X
X    /* Make it printable on remote machine */
X    sendline(015); sendline(012);
X    /*
X     * Uncork the remote in case a fake XOFF has stopped data flow
X     */
X    if (type != ZFIN)
X         sendline(021);
X    flush_modem();
X}
X
X/*
X * Send binary array buf of length length, with ending ZDLE sequence frameend
X */
Xzsdata(buf, length, frameend)
Xregister char *buf;
Xregister int length, frameend;
X{
X    register unsigned int crc;
X
X    vfile("zsdata: length=%d end=%x", length, frameend);
X    if (Txfcs32)
X         zsda32(buf, length, frameend);
X    else {
X         crc = 0;
X         for (;--length >= 0;) {
X              zsendline(*buf);
X              crc = updcrc((0377& *buf++), crc);
X         }
X         xsendline(ZDLE); xsendline(frameend);
X         crc = updcrc(frameend, crc);
X
X         crc = updcrc(0,updcrc(0,crc));
X         zsendline(crc>>8); zsendline(crc);
X    }
X    if (frameend == ZCRCW) {
X         xsendline(XON);   flush_modem();
X    }
X}
X
Xzsda32(buf, length, frameend)
Xregister char *buf;
Xregister int length, frameend;
X{
```

```
X    register unsigned long crc;
X
X    crc = 0xFFFFFFFFL;
X    for (;--length >= 0;) {
X         zsendline(*buf); crc = UPDC32((0377& *buf++), crc);
X    }
X    xsendline(ZDLE); xsendline(frameend);
X    crc = UPDC32(frameend, crc);
X
X    crc = ~crc;
X    for (length=4; --length >= 0;) {
X         zsendline((unsigned int)(crc&0xFFFF));  crc >>= 8;
X    }
X}
X
X/*
X * Receive array buf of max length with ending ZDLE sequence
X *  and CRC.  Returns the ending character or error code.
X */
Xzrdata(buf, length)
Xregister char *buf;
Xregister int length;
X{
X    register int c;
X    register unsigned int crc;
X    register int d;
X
X    if (Rxframeind == ZBIN32)
X         return zrdat32(buf, length);
X
X    crc = Rxcount = 0;
X    for (;;) {
X         if ((c = zdlread()) & ~0377) {
Xcrcfoo:
X              switch (c) {
X              case GOTCRCE:
X              case GOTCRCG:
X              case GOTCRCQ:
X              case GOTCRCW:
X                   crc = updcrc((d=c)&0377, crc);
X                   if ((c = zdlread()) & ~0377)
X                        goto crcfoo;
X                   crc = updcrc(c, crc);
X                   if ((c = zdlread()) & ~0377)
X                        goto crcfoo;
X                   crc = updcrc(c, crc);
X                   if (crc & 0xFFFF) {
X                        log2("Bad data CRC %x", crc);
X                        return ERROR;
X                   }
X                   vfile("zrdata: cnt = %d ret = %x", Rxcount, d);
X                   return d;
X              case GOTCAN:
```

```
X                 log2("ZMODEM: Sender Canceled");
X                 return ZCAN;
X             case TIMEOUT:
X                 log2("ZMODEM data TIMEOUT");
X                 return c;
X             default:
X                 log2("ZMODEM bad data subpacket ret=%x", c);
X                 return c;
X             }
X         }
X         if (--length < 0) {
X             log2("ZMODEM data subpacket too long");
X             return ERROR;
X         }
X         ++Rxcount;
X         *buf++ = c;
X         crc = updcrc(c, crc);
X         continue;
X    }
X}
Xzrdat32(buf, length)
Xregister char *buf;
Xregister int length;
X{
X    register int c;
X    register unsigned long crc;
X    register int d;
X
X    crc = 0xFFFFFFFFL;  Rxcount = 0;
X    for (;;) {
X         if ((c = zdlread()) & ~0377) {
Xcrcfoo:
X             switch (c) {
X             case GOTCRCE:
X             case GOTCRCG:
X             case GOTCRCQ:
X             case GOTCRCW:
X                 crc = UPDC32((d=c)&0377, crc);
X                 if ((c = zdlread()) & ~0377)
X                     goto crcfoo;
X                 crc = UPDC32(c, crc);
X                 if ((c = zdlread()) & ~0377)
X                     goto crcfoo;
X                 crc = UPDC32(c, crc);
X                 if ((c = zdlread()) & ~0377)
X                     goto crcfoo;
X                 crc = UPDC32(c, crc);
X                 if ((c = zdlread()) & ~0377)
X                     goto crcfoo;
X                 crc = UPDC32(c, crc);
X                 if (crc != 0xDEBB20E3L) {
X                     log2("Bad data CRC");
X                     return ERROR;
```

```
X                    }
X                    vfile2("zrdat32: cnt = %ld ret = %x", Rxcount, d);
X                    return d;
X               case GOTCAN:
X                    log2("ZMODEM: Sender Canceled");
X                    return ZCAN;
X               case TIMEOUT:
X                    log2("ZMODEM data TIMEOUT");
X                    return c;
X               default:
X                    log2("ZMODEM bad data subpacket ret");
X                    return c;
X               }
X          }
X          if (--length < 0) {
X               log2("ZMODEM data subpacket too long");
X               return ERROR;
X          }
X          ++Rxcount;
X          *buf++ = c;
X          crc = UPDC32(c, crc);
X          continue;
X     }
X}
X
X
X/*
X * Read a ZMODEM header to hdr, either binary or hex.
X *  eflag controls local display of non zmodem characters:
X *  0:  no display
X *  1:  display printing characters only
X *  2:  display all non ZMODEM characters
X *  On success, set Zmodem to 1 and return type of header.
X *   Otherwise return negative on error
X */
Xzgethdr(hdr, eflag)
Xregister char *hdr;
Xregister int eflag;
X{
X    register int c, n, cancount;
X
X    n = Baudrate; /* Max characters before start of frame */
X    cancount = 5;
Xagain:
X    Rxframeind = Rxtype = 0;
X    switch (c = noxread7()) {
X    case RCDO:
X    case TIMEOUT:
X         goto fifi;
X    case CAN:
X         if (--cancount <= 0) {
X              c = ZCAN; goto fifi;
X         }
```

```
X    /* **** FALL THRU TO **** */
X    default:
Xagn2:
X         if ( --n == 0) {
X              log2("ZMODEM Garbage count exceeded");
X              return(ERROR);
X         }
X         if (eflag && ((c &= 0177) & 0140))
X              bttyout(c);
X         else if (eflag > 1)
X              bttyout(c);
X         if (c != CAN)
X              cancount = 5;
X         goto again;
X    case ZPAD:         /* This is what we want. */
X         break;
X    }
X    cancount = 5;
Xsplat:
X    switch (c = noxread7()) {
X    case ZPAD:
X         goto splat;
X    case RCDO:
X    case TIMEOUT:
X         goto fifi;
X    default:
X         goto agn2;
X    case ZDLE:         /* This is what we want. */
X         break;
X    }
X
X    switch (c = noxread7()) {
X    case RCDO:
X    case TIMEOUT:
X         goto fifi;
X    case ZBIN:
X         Rxframeind = ZBIN;
X         c =  zrbhdr(hdr);
X         break;
X    case ZBIN32:
X         Rxframeind = ZBIN32;
X         c =  zrbhdr32(hdr);
X         break;
X    case ZHEX:
X         Rxframeind = ZHEX;
X         c =  zrhhdr(hdr);
X         break;
X    case CAN:
X         if (--cancount <= 0) {
X              c = ZCAN; goto fifi;
X         }
X         goto agn2;
X    default:
```

```
X         goto agn2;
X    }
X    Rxpos = hdr[ZP3] & 0377;
X    Rxpos = (Rxpos<<8) + (hdr[ZP2] & 0377);
X    Rxpos = (Rxpos<<8) + (hdr[ZP1] & 0377);
X    Rxpos = (Rxpos<<8) + (hdr[ZP0] & 0377);
Xfifi:
X    switch (c) {
X    case GOTCAN:
X         c = ZCAN;
X    /* **** FALL THRU TO **** */
X    case ZNAK:
X    case ZCAN:
X    case ERROR:
X    case TIMEOUT:
X    case RCDO:
X         vfile2("ZMODEM: Got %s %s", frametypes[c+FTOFFSET],
X           (c >= 0) ? "header" : "error");
X    /* **** FALL THRU TO **** */
X    default:
X         if (c >= -3 && c <= FRTYPES)
X              vfile2("zgethdr: %s %lx", frametypes[c+FTOFFSET], Rxpos);
X         else
X              vfile2("zgethdr: %ld %lx", (long)c, Rxpos);
X    }
X    return c;
X}
X
X/* Receive a binary style header (type and position) */
Xzrbhdr(hdr)
Xregister char *hdr;
X{
X    register int c, n;
X    register unsigned int crc;
X
X    if ((c = zdlread()) & ~0377)
X         return c;
X    Rxtype = c;
X    crc = updcrc(c, 0);
X
X    for (n=4; --n >= 0;) {
X         if ((c = zdlread()) & ~0377)
X              return c;
X         crc = updcrc(c, crc);
X         *hdr++ = c;
X    }
X    if ((c = zdlread()) & ~0377)
X         return c;
X    crc = updcrc(c, crc);
X    if ((c = zdlread()) & ~0377)
X         return c;
X    crc = updcrc(c, crc);
X    if (crc & 0xFFFF) {
```

```
X         log2("Bad Header CRC"); return ERROR;
X    }
X    Zmodem = 1;
X    return Rxtype;
X}
X
X/* Receive a binary style header (type and position) with 32 bit FCS */
Xzrbhdr32(hdr)
Xregister char *hdr;
X{
X    register int c, n;
X    register unsigned long crc;
X
X    if ((c = zdlread()) & ~0377)
X         return c;
X    Rxtype = c;
X    crc = 0xFFFFFFFFL; crc = UPDC32(c, crc);
X
X    for (n=4; --n >= 0;) {
X         if ((c = zdlread()) & ~0377)
X              return c;
X         crc = UPDC32(c, crc);
X         *hdr++ = c;
X    }
X    for (n=4; --n >= 0;) {
X         if ((c = zdlread()) & ~0377)
X              return c;
X         crc = UPDC32(c, crc);
X    }
X    if (crc != 0xDEBB20E3L) {
X         log2("Bad Header CRC"); return ERROR;
X    }
X    Zmodem = 1;
X    return Rxtype;
X}
X
X
X/* Receive a hex style header (type and position) */
Xzrhhdr(hdr)
Xchar *hdr;
X{
X    register int c;
X    register unsigned int crc;
X    register int n;
X
X    if ((c = zgethex()) < 0)
X         return c;
X    Rxtype = c;
X    crc = updcrc(c, 0);
X
X    for (n=4; --n >= 0;) {
X         if ((c = zgethex()) < 0)
X              return c;
```

```
X         crc = updcrc(c, crc);
X         *hdr++ = c;
X    }
X    if ((c = zgethex()) < 0)
X         return c;
X    crc = updcrc(c, crc);
X    if ((c = zgethex()) < 0)
X         return c;
X    crc = updcrc(c, crc);
X    if (crc & 0xFFFF) {
X         log2("Bad Header CRC"); return ERROR;
X    }
X    if (readline(1) == '\r')    /* Throw away possible cr/lf */
X         readline(1);
X    Zmodem = 1; return Rxtype;
X}
X
X/* Send a byte as two hex digits */
Xzputhex(c)
Xregister int c;
X{
X    static char   digits[] = "0123456789abcdef";
X
X    if (Verbose>4)
X         vfile("zputhex: %x", c);
X    sendline(digits[(c&0xF0)>>4]);
X    sendline(digits[(c)&0xF]);
X}
X
X/*
X * Send character c with ZMODEM escape sequence encoding.
X *  Escape XON, XOFF. Escape CR following @ (Telenet net escape)
X */
Xzsendline(c)
Xregister int c;
X{
X    static int lastsent;
X
X    switch (c & 0377) {
X    case ZDLE:
X         xsendline(ZDLE);
X         xsendline (lastsent = (c ^= 0100));
X         break;
X    case 015:
X    case 0215:
X         if ((lastsent & 0177) != '@')
X              goto sendit;
X    /* **** FALL THRU TO **** */
X    case 020:
X    case 021:
X    case 023:
X    case 0220:
X    case 0221:
```

```
X    case 0223:
X    if(SendType)
X    {
X         if (Zctlesc<0)
X              goto sendit;
X    }
X
X         xsendline(ZDLE);
X         c ^= 0100;
Xsendit:
X         xsendline(lastsent = c);
X         break;
X    default:
X         if(SendType)
X         {
X              if (Zctlesc>0 && ! (c & 0140)) {
X                   xsendline(ZDLE);
X                   c ^= 0100;
X              }
X         }
X
X         xsendline(lastsent = c);
X    }
X}
X
X/* Decode two lower case hex digits into an 8 bit byte value */
Xzgethex()
X{
X    register int c;
X
X    c = zgeth1();
X    if (Verbose>4)
X         vfile("zgethex: %x", c);
X    return c;
X}
Xzgeth1()
X{
X    register int c, n;
X
X    if ((c = noxread7()) < 0)
X         return c;
X    n = c - '0';
X    if (n > 9)
X         n -= ('a' - ':');
X    if (n & ~0xF)
X         return ERROR;
X    if ((c = noxread7()) < 0)
X         return c;
X    c -= '0';
X    if (c > 9)
X         c -= ('a' - ':');
X    if (c & ~0xF)
X         return ERROR;
```

```
X    c += (n<<4);
X    return c;
X}
X
X/*
X * Read a byte, checking for ZMODEM escape encoding
X *  including CAN*5 which represents a quick abort
X */
Xzdlread()
X{
X    register int c;
X
X    if ((c = readline(Rxtimeout)) != ZDLE)
X        return c;
X    if ((c = readline(Rxtimeout)) < 0)
X        return c;
X    if (c == CAN && (c = readline(Rxtimeout)) < 0)
SHAR_EOF
echo "End of part 7"
echo "File ZM.C is continued in part 8"
echo "8" > s2_seq_.tmp
exit 0


From 964259832201967438
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,932c4d92e0d9fb49,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-05-30 09:17:06 PST
Path: utzoo!attcan!uunet!husc6!mailrus!rutgers!ucsd!ucbvax!agate!saturn!ssyx!
koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i048: zmdm -- terminal program with file transfer  part08/08
Keywords: mwc, alcyon, shar
Message-ID: <3514@saturn.ucsc.edu>
Date: 30 May 88 16:17:06 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1089
Approved: koreth@ssyx.ucsc.edu

Submitted-by: bammi@mandrill.ces.cwru.edu (Jwahar R. Bammi)
Posting-number: Volume 1, Issue 48
Archive-name: zmdm/part08

#!/bin/sh
# this is part 8 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file ZM.C continued
#
CurArch=8
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
```

```
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file ZM.C"
sed 's/^X//' << 'SHAR_EOF' >> ZM.C
X        return c;
X    if (c == CAN && (c = readline(Rxtimeout)) < 0)
X        return c;
X    if (c == CAN && (c = readline(Rxtimeout)) < 0)
X        return c;
X    switch (c) {
X    case CAN:
X        return GOTCAN;
X    case ZCRCE:
X    case ZCRCG:
X    case ZCRCQ:
X    case ZCRCW:
X        return (c | GOTOR);
X    case ZRUB0:
X        return 0177;
X    case ZRUB1:
X        return 0377;
X    default:
X        if ((c & 0140) ==  0100)
X            return (c ^ 0100);
X        break;
X    }
X    log2("Got bad ZMODEM escape sequence %x", c);
X    return ERROR;
X}
X
X/*
X * Read a character from the modem line with timeout.
X *  Eat parity, XON and XOFF characters.
X */
Xnoxread7()
X{
X    register int c;
X
X    for (;;) {
X        if ((c = readline(Rxtimeout)) < 0)
X            return c;
X        switch (c &= 0177) {
X        case XON:
X        case XOFF:
X            continue;
X        default:
X            return c;
X        }
```

```
X    }
X}
X
X/* Store long integer pos in Txhdr */
Xstohdr(pos)
Xregister long pos;
X{
X    Txhdr[ZP0] = pos;
X    Txhdr[ZP1] = pos>>8;
X    Txhdr[ZP2] = pos>>16;
X    Txhdr[ZP3] = pos>>24;
X}
X
X/* Recover a long integer from a header */
Xlong rclhdr(hdr)
Xregister char *hdr;
X{
X    register long l;
X
X    l = (hdr[ZP3] & 0377);
X    l = (l << 8) | (hdr[ZP2] & 0377);
X    l = (l << 8) | (hdr[ZP1] & 0377);
X    l = (l << 8) | (hdr[ZP0] & 0377);
X    return l;
X}
X
X/* -eof- */
SHAR_EOF
echo "File ZM.C is complete"
chmod 0600 ZM.C || echo "restore of ZM.C fails"
echo "x - extracting ZMDM.DOC (Text)"
sed 's/^X//' << 'SHAR_EOF' > ZMDM.DOC &&
X    zmdm.doc, v1.6 ++jrb
X
X                ACKNOWLEDGEMENTS
X
X    ZMDM was derived from rz/sz for Unix  posted by
X    Chuck Forsberg (...!tektronix!reed!omen!caf ). We
X    thank him for his excellent code, and for giving
X    us permission to use and distribute his code and
X    documentation.
X
X    The code for detecting baud rate at startup is courtesy
X    of Brian Katzung (katzung@laidbak.UUCP). Thanks very
X    much!
X
X    Andy Nicola did a lot of testing of the new features.
X    Thank You very much!
X
X    Thanks to the many users on the net who wrote in.
X
X-------------------------------------------------------------------------
X    Enhancements since V1.2: (the last Usenet  release).
```

X
X o Some cleanup, moved common things around.
X   Some cosmetic additions.
X
X o MWC 3.0 compatible. The sample MW makefiles
X   reflect the setup required for MWC V 3.0.6.
X
X o Manx Aztec C compatible. Tested with V 3.6a
X   of the compiler. Produces the smallest code
X   of all the compilers tested. See config.h,
X   makefile.man and makefman.sta.
X   CAUTION: for MegaST and 4Meg ST owners--
X   The Manx Version will not work correctly
X   for you due to a bug in the start up file
X   -- we are still trying to figure out exactly
X   what the problem is!
X
X o Auto Baud Rate detection on startup:
X   Previously ZMDM would set the baud rate
X   to a compile time configurable rate on
X   startup (it needed to know the baud rate
X   to determine its default packet size). Thanks
X   to code contributed by Brian Katzung, now it
X   detects the baud rate on startup.
X   Of course you can set the baud rate
X   within ZMDM (by hitting '<HELP>' 'b').
X
X o Compile time option to use all available memory
X   as recv/send buffer. See DYNABUF, LEAVEALONE and
X   MINACC preprocessor symbols in config.h. If you
X   prefer to use a fixed size buffer, this is still
X   possible by not #define'ing DYNABUF and setting
X   BBUFSIZ. MegaST OK - thanks Andy Nicola.
X
X o Stand Alone versions of RZ and SZ can be made
X   by compiling the sources with the preprocessor
X   symbols 'STANDALONE' #define'ed. See the makefile
X   'MAKEFILE.STA' for more details.
X   Stand Alone versions DO NOT do any command line
X   argument expansions for wild cards or directories
X   (unlike the integrated ZMDM).
X   Stand Alone versions tested from within Gulam,
X   Flash and Intersect. Thanks to Andy Nicola for
X   testing these features.
X
X o Remote versions of ZMDM, RZ and SZ can be made by
X   compiling the sources with the preprocessor symbol
X   'REMOTE' #define'ed. A remote version listens and
X   does all its I/O thru the serial port. Of course
X   it has to be fired up from console. One of our
X   area BBS's provides this as a D/L option, where
X   the BBS program execs a remote ZMDM, and one you exit
X   you return back to the BBS program.

```
X
X           o Added -B (note: uppercase B) option to both RZ and SZ.
X         the -B function is a binary mode override, that disregards
X         the extension when deciding the transfer mode. This is
X         useful when say you are backing up your whole disk
X         to say a Unix host. (backup the exact image of the files),
X         or doing St-to-St type transfers.
X
X       o Now compatible with the PD DLIB library. The
X         pre-processor symbol DLIBS must be defined. Only
X         tested with the Alcyon version of the library.
X
X       o Added the phone module (a simple dialer with a
X         telephone numbers directory). This module is
X         only compiled in if the preprocessor symbol
X         'PHONES' is #define'ed. For those of you who
X         are familiar with XMDM, this module was directly
X         ripped out of there, with a small bug fix that
X         prevented it from running under TurboSt.ACC.
X
X       o Long packet lengths now allowed at slower baud rates
X         rather than insisting on using the default as the max
X         packet length. For instance
X            sz -l 1024 files..
X         will now send 1024 byte packets at low baud rates (<= 2400).
X         The corresponding change has been made to the unix
X         end of the software too.
X
X       o TurboSt.ACC note:
X            The high rez toggle (25/50 lines) on a
X       monochrome monitor will NOT work is TurboSt.ACC is
X       installed. The problem is on the TurboSt.ACC end. Using
X       the high rez toggle does'nt cause any ill-effect, it just
X       does not do anything.
X
X   Enhancements since V1.0:
X       o (This enhancement is only present when
X          you compile with the pre-processor symbol
X          `RECURSE' defined - see the makefile's)
X         sz now takes a  directory as an argument.
X         If the name of a directory is given as an
X         argument, then the contents of that directory
X         and all its subdirectories are sent. A new
X         option to sz is '-P <dir or file>'
X         (capital 'P' not 'p'), that
X         'prunes' the expansion of a directory. For
X         example if you wanted to send all the files
X         in the 'C' partition of your hard disk, except
X         the 'tmp', 'usr' and 'foo\bar' subdirectories, you would
X         issue the following command in the transfer
X         shell:
X
X            sz -f -P c:\tmp -P c:\usr -P c:\foo\bar c:\
```

```
X
X          Notice that you may specify multiple -P 'rune
X          options. The argument to the -P option may also be
X          the name of a file that you want to prune off.
X          Also note that to send full path names
X          to the remote end, the '-f' option is required,
X          otherwise the remote end will create all the
X          files in its current working directory. (Also
X          see CWRU extension to Unix 'rz', where 'rz' when
X          receiving full path names (ie. when you specify the
X          '-f' option to sz on the ST end) will create all
X          subdirectories required to receive the file). The
X          ST 'rz' always creates all subdirectories required
X          to receive a path when the other end is sz'ing with
X          the '-f' option. Also note that in both the rz's
X          all subdirectories are created relative to the
X          current working directory, even though the path may
X          specify an absolute path.
X
X           o File name mapping is slightly changed from V1.0.
X          When receiving a filename that has multiple '.'s
X          all but the last '.' is replace with an '_'. So
X          'foo.bar.ext' becomes 'foo_bar.ext'. Similarly
X          '123.456.789\aaa.bbb.ccc' becomes '123_456.789\aaa_bbb.ccc'.
X          As in V1.0, the filename and extension (of each
X          component of a path name) are truncated to 8 and 3
X          characters respectively. So '123.456789.ext\foobarbaz'
X          becomes '123_4567.ext\foobarba'. Also note that
X          while sending, '\'s are sent as '/'s. The ST drive
X          specifier part of a path name is never sent.
X          So 'c:\file.ext' is sent as '/file.ext' (remember:
X          that full pathnames are only sent when the '-f'
X          option is specified to 'sz').
X
X        o Many bug fixes since V1.0!
X
X------------------------------------------------------------------------
X
X
X    Zmdm consists of two main components:
X         o The terminal emulator
X         o The transfer shell
X
X    - The terminal emulator emulates a 80*25 terminal
X    (or optionally the 80*50 mode on Mono systems only).
X    It uses the escape codes of the bios built in enhanced
X    vt52 emulator. For UN*X users i have included the
X    termcap entry that we use.
X
X    - The transfer shell lets you send/receive files using
X    Xmodem/Xmodem-CRC/Xmodem-1K/Ymodem or Zmodem protocols.
X         In addition it provides UN*X csh like command for your
X    convenience, and does (TOS style) wild-card handling,
```

```
X        and (only single) quoting of arguments. Please note that
X    this is a convenience feature, and in no way does it
X    pretend to be compatible with csh or any other shell.
X    See the accompanying file YMODEM.DOC by Chuck Forsberg,
X    for details of the above transfer protocols.
X
X    This note describes the commands available from the
X    "transfer shell". The transfer shell is the built in command line
X    interface. In case you are wondering why we choose to
X    use a command line interface, the answer is quite simple.
X    The functions available have so many options and combinations
X    thereof, that it would be totally impractical if not impossible
X    to put them in menus/dialogues (each menu will have to be at least
X    3 levels deep). Of course it is much easier to type something
X    like 'sz -l 128 -L 128 *.c' than answering the at least three
X    dialogues required to pick up the same info. You enter the
X    transfer shell from the terminal emulator by hitting <HELP>
X    and choosing 'T' from the main menu. Once you are done using the
X    transfer shell, you simply hit <RETURN> to pop back into the terminal
X    emulator, exactly where you left off.
X
X    This program was written primarily for situations where you
X    are connected to remote hosts either directly or via modems.
X    The timing and error parameters are NOT suitable for hosts
X    such as CompuServe etc, where there may large delays or your
X    phone line is noisy. We distributed a program called XMDM earlier
X    that is very tolerant of delays/noisy lines and has been used
X    very successfully for BBS'ing around the country. XMDM has
X    conveniences such as dialing directories etc for that purpose.
X    If you have a old version of XMDM ( < V1.8) that does not have
X    dialing directories etc, please mail me for a new copy.
X
X    ZMDM has been tested in the following ways:
X    1) Between two ST's running ZMDM. 19200 Baud no problem!
X    2) With unix sz/rz running on a BSD4.3 on Vaxen.
X    3) With unix sz/rz running on Apollo Dn330/Dn300 under Aegis and
X       Domain IX (over a siologin line).
X    4) With unix sz/rz running on a Sun under SunOS.
X    6) With an Ibm At running Procomm using Ymodem/Xmodem.
X    7) With ForemSt bbs, with the BBS sending in Ymodem batch
X       mode. (note: ForemSt bbs's do not take batch uploads -
X       Matt Singer are you listening??)
X
X    8) With a Mac running MicroPhone using Ymodem mode.
X    9) With an Intel 210 box running Xenix (yech!).
X
X    UNIX users please note:
X        If the load on your system is high, the UNIX tty
X    driver drops characters when receiving at 9600 or higher baud rates.
X    There are two ways around this problem:
X    - use shorter packet and frame lengths using the `-l' and `-L' options
X    of SZ. For examples `sz -l 128 -L 128 files...' work most of
X    the time at 9600/19200 baud without any problem.
```

```
X
X    - reduce your baud rate to 4800 baud. Normally the first solution
X    works fine, but if you have an extraordinarily loaded machine
X    (or are running on a brain damaged 750 with DZ11's), then 4800 baud
X    works better.
X
X    - Those who have 3b2 boat anchors, the tty driver (at least on
X     the boxes we have) is totally useless over 1200 baud.
X
X    -----
X
X
XNAME
X     rb, rz - XMODEM, YMODEM, ZMODEM (Batch) file receive
X
XSYNOPSIS
X     rz [-Bpqtv]
X     rb [-pqtv]
X     rz [-cqtv] file
X
X
XDESCRIPTION
X     This program uses error correcting  protocol to receive files
X     over a serial port    from a variety of programs running under
X     TOS, PC-DOS, CP/M, Unix, and other operating systems.
X
X     The first form of rz (Receive ZMODEM) receives files with
X     the ZMODEM   batch protocol.     If the   sending   program
     does not
X     support ZMODEM, rz    steps down to YMODEM protocol after 50
X     seconds.  This delay can be eliminated by calling the pro-
X     gram as rb   .
X
X     When receiving with XMODEM or YMODEM, Rz accepts either
X     standard 128 byte sectors or 1024 byte sectors (YAM -k
X     option).  The user    should determine when the longer block
X     length actually improves throughput without causing prob-
X     lems.
X
X     If extended file information (file  length,   etc.) is
X     received, the file    length controls    the number of bytes writ-
X     ten to the   output dataset (YMODEM only), and the modify time
X     and file mode (iff    non zero) are set accordingly. A special feature
X     of the ST implementation is that if the remote end sends
X     full path names (-f option of sz) all required directories
X     will be automatically created on the ST end. The directories
X     will be rooted at the current directory and not at the root level.
X
X     The second   form of   rz receives a single file with XMODEM
X     protocol.    The user must supply the file name to both send-
X     ing and receiving programs.
X
X     When rz is invoked, Verbose is set to 2, causing frame by
```

```
X     frame progress reports to stderr.  This may be disabled with
X     the q option.
X
X
X     The meanings of the available options are:
X
X     B    (ZMODEM) force all files to be received in binary mode.
X      Useful for ST-to-ST transfers (for Ascii files where
X      you don't want LF to CRLF conversion). This is
X      a local override, specifying `-B' overrides any mode
X      the sender specifies (it however will still honor any `protect'
X      or `append' requests from the sender)
X      (Applicable to ZMODEM transfers only).
X
X     c    Request 16 bit CRC.  XMODEM file transfers default to
     8
X      bit checksum.     YMODEM   and ZMODEM normally use 16 bit
X      CRC.
X
X     p    (ZMODEM) Protect: skip file if destination file exists.
X
X     q    Quiet  suppresses verbosity.
X
X     t tim
X      Change timeout to tim tenths of seconds.
X
X     v    Verbose, more v's give more verbose. Info is also
X      appended to logfile 'rzlog' .  More v's generate more output.
X
XZMODEM CAPABILITIES
X     Rz  supports incoming ZMODEM binary  (-b), ASCII (-a), protect
X     (-p), and append (-+) requests, and ZMODEM   command
     execu-
X     tion. The incoming mode may be optionally overriden by specifying
X     the `-B' option to Rz, in which case all files will be received
X     in binary mode regardless of the incoming mode (protect and append will
X     still be obeyed).
X
X     Rz also supports incoming pathnames, and will recursively
X     create all subdirectories as required. Rz preserve file mod
X     times, and file protection. The translation of Unix protection
X     bits to ST file attributes is based on the Unix owner (07XX)
X     protection bits.  Presently, only read and write bits are considered.
X     All received pathnames are considered to be rooted at the
X     current working directory. Any leading '/' from unix systems
X     is discarded (ie. all incoming pathnames are considered unrooted,
X     and are locally 'planted' at the current working directory).
X     Ascii/Binary file modes are determined by examining the file
X     extension on the incoming path name. If the extension is one
X     of the following (case independent), then it is assumed to be
X     a binary file, ascii other wise. The ascii translation of
X     '\n' to '\r\n' is done locally for ZMODEM (only).
X     For X or Y modem the sender determines the file type (see
```

```
X     the file 'common.c' for the latest list).
X
X    ".PRG", ".TOS", ".TTP", ".ARC", ".ACC", ".IMG", ".RSC", ".O",
X        ".OBJ", ".NEO", ".PIC", ".PI1", ".PI2", ".PI3", ".PQ1", ".PQ2",
X        ".PQ3", ".BRD", ".ANI", ".STW", ".FNT", ".PRT", ".SNG", ".NEC",
X        ".CNF", ".Z"  , ".DFN", ".GEM", ".EZD", ".LNK", ".SYM",
X        ".PIX", ".X32", ".OUT", ".A",   ".CCC", ".CL",  ".CMD", ".COM",
X        ".CRL", ".DAT", ".DIR", ".EXE", ".OVL", ".PAG", ".REL", ".SAV",
X        ".SUB", ".SWP", ".SYS", ".TAR", ".UTL", ".IM",  ".PAK"
X
X
X
XRZ/SZ(1)           ST Programmer's Manual              RZ/SZ(1)
X
XNAME
X     sz
X
XSYNOPSIS
X     sz [-+BdefkLlNnopqtuvy][-P <directory of file>]*  file ...
X     sz -X [-kqtuv] file
X     sz [-oqtv] -c COMMAND
X     sz [-oqtv] -i COMMAND
X
XDESCRIPTION
X     Sz uses the ZMODEM, YMODEM or XMODEM error    correcting proto-
X     col to send one or    more files over    a serial port to a
X     variety of   programs running under PC-DOS, CP/M, Unix, VMS, TOS
X     and other operating systems.
X
X
X     The first form of sz sends one or more files with ZMODEM or
X     YMODEM batch protocol.  Normally, only the   file name part of
X     the pathname is transmitted. Additional
X     information about the file is transmitted.     If the
     receiving
X     program uses this information, the  transmitted file length
X     controls the exact    number of bytes    written  to the output
X     dataset, and the modify time and file mode    are set
     accord-
X     ingly.
X
X     The second   form of   sz uses   the -X flag to send a single file
X     with XMODEM or XMODEM-1k protocol.   The user must supply the
X     file name to both sending and receiving programs.
X
X
X     The third form sends a single COMMAND to the receiver for
X     execution.    Sz exits with the COMMAND return value.
X
X
X     The fourth   form sends a single COMMAND to the receiver for
X     execution.    Sz exits as soon as the receiver has correctly
X     received the command, before it is  executed.
```

X
X
X   In SZ Verbose is set to 2, causing frame by frame pro-
X   gress reports to stderr. This may be disabled with the q
X   option.
X
X   The meanings of the available options are:
X
X   +   Instruct the receiver to append transmitted data to an
X   existing file   (ZMODEM   only).
X
X   B   (ZMODEM) force all files to be sent in binary mode.
X   Useful for ST-to-ST transfers (for Ascii files where
X   you don't want LF to CRLF conversion).
X   (Applicable to ZMODEM transfers only).
X
X   c COMMAND
X   Send COMMAND to the receiver for execution, return with
X   COMMAND's exit status.
X
X   d   Change all instances of "." to "/" in   the transmitted
X   pathname. Thus, C.omenB0000 (which is unacceptable to
X   MSDOS or CP/M) is transmitted as C/omenB0000.   If the
X   resultant filename has more than 8 characters   in the
X   stem, a "." is inserted to allow a total of eleven.
X
X   E   Escape only Ctrl-X control characters; normally XON,
X   XOFF, CR-@-CR, and Ctrl-X are escaped.
X
X   e   Escape all control characters; normally XON, XOFF, CR-
X   @-CR, and Ctrl-X are escaped.
X
X   f   Send Full pathname. Normally directory prefixes are
X   stripped from   the transmitted   filename.
X
X   i COMMAND
X   Send COMMAND to the receiver for execution, return
X   Immediately upon the receiving program's successful
X   reception of the command.
X
X   k   (XMODEM/YMODEM) Send files using 1024   byte blocks
X   rather than the default 128 byte blocks. 1024 byte
X   packets speed   file transfers at high bit rates. (ZMO-
X   DEM streams the data for the best possible throughput.)
X
X   L N Use ZMODEM sub-packets of length N. A larger
     N (32 <=
X   N <= 1024) gives slightly higher throughput, a smaller
X   N speeds error recovery. The default is 128 below 300
X   baud, 256 above 300 baud, or 1024 above 2400 baud.
X
X   l N Wait for the receiver to acknowledge correct data every
X   N (32 <= N <= 1024) characters. This may be used to

```
X      avoid  network  overrun  when XOFF flow control is lack-
X      ing.
X
X     n    (ZMODEM) Send   each file if destination file does not
X      exist.  Overwrite destination  file if  source file is
X      newer  or longer than the destination file.
X
X     N    (ZMODEM) Send   each file if destination file does not
X      exist.  Overwrite destination  file if  source file has
X      different length or date.
X
X     o    (ZMODEM) Disable automatic selection of 32 bit CRC.
X
X     p    (ZMODEM) Protect existing destination   files by skipping
X      transfer if the destination file exists.
X
X     q    Quiet  suppresses verbosity.
X
X     r    Resume interrupted file transfer.  If   the source file
X      is longer than the destination file, the transfer com-
X      mences at the   offset in the source file that equals the
X      length of the   destination file.
X
X     t tim
X      Change timeout to tim tenths of seconds.
X
X     u    Unlink the file after successful transmission.
X
X     v    Verbose causes a list of file  names to be appended to
X      szlog .More v's generate more output.
X
X     X    Send a single   file with XMODEM or XMODEM-1k protocol.
X
X     y    Instruct a ZMODEM receiving program to overwrite any
X      existing file   with the same name.
X
X     P <directory or file>
X    (ZMODEM) Sz may be given the name of
X    a directory, in which case it will send the contents
X    of the directory, and all the subdirectories thereof.
X    The -P 'Prune' option is used to prune out the named directory
X    or file while expanding the directory tree. Multiple
X    -P options may be specified. This option
X    is only present when the source is compiled with the
X    pre-processor symbol `RECURSE' defined.
X
X    SZ guesses the file mode by examining the filename extension as
X    discusses under RZ above. SZ -f send full pathnames with '\'
X    converted to '/'. The ST drive specifier is never sent.
X
XSEE ALSO
X     ZMODEM.DOC, YMODEM.DOC, IMP(CP/M),  cu(1), Professional-YAM
X     manual, sz(omen), usq(omen), undos(omen)
```

```
X
X     Compile time options required for various operating systems
X     are described in the source file.
X
XNOTES
X    Depending on your file structure, when using the recursively
X    descend a directory option of Sz, ZMDM can demand a lot of
X    dynamic memory. When the pre-processor symbol `RECURSE' is
X    defined, for Mark Williams C, `_stksize' is set to 16K. If
X    you are using Alcyon C, please use a decent version of
X    GEMSTART.S. We use the version from pratt@atari with
X    memory model (STACK=1), which gives quarter of available
X    memory to stack+heap. Note that all the dynamic memory is
X    taken off the program heap (for Alcyon, and off the Arena
X    above the stack for Mark Wiliams), ie. ZMDM uses `malloc' not
X    `Malloc'.
X
X    It is possible to run out of dynamic memory when descending
X    a file structure with a lot of files and/or deep-hierarchies.
X    The simplest way to work around this problem is to do the
X    sending in parts by using the -P rune option of Sz. For
X    instance if you run out of memory while trying to send your
X    C partition, you may want to prune out some directories and
X    send things in parts. In my case i backup my C partition to
X    our unix host in the following manner:
X
X    sz -f -P c:\bin -P c:\lib c:\  /* send eveything in the C
X                          partition except the 'bin'
X                          and 'lib' subdirectories */
X
X    sz -f c:\bin c:\lib           /* then send the 'bin' and 'lib'
X                          directories */
X
X    The second solution depends on if you are using Mark Williams
X    or Alcyon C libraries. If you are running out of memory because
X    of deep hierarchies, you need to increase the stack. If you are
X    running out of memory because of a lot of files, then you
X    need to decrease the stack to allow for a larger arena in
X    the case of MWC (and trade-off for depth), or in the case of
X    Alcyon you need to increase stack+heap. On systems with a
X    large Ramdisk or many ACCs make sure you have enough space
X     before doing this.
X
X    The TTY input buffering on  some systems may not allow long
X    blocks or streaming input,  especially at high baud rates.
X    The Pro-YAM zmodem l numeric parameter may be set to a value
X    between 64    and 1024 to limit the burst length.
X
XBUGS
X     THIS SOFTWARE IS STILL UNDER DEVELOPMENT AND ALMOST POSITIVELY
X     CONTAINS BUGS. PLEASE REPORT ALL SUCH CRITTERS TO mandrill!bammi.
X
X     Pathnames are restricted to 127 characters.  In XMODEM sin-
```

```
X     gle file mode, the    pathname given on the command line is
X     still processed as    described above.
X
XFILES
X
X     (rz|sz)log   stores debugging output generated with -vvv[v]*
X           option.
X
XOTHER COMMANDS
X    The following command are available in the
X    transfer shell:
X
X    <regular-expression> ::= <file name> | <wild card> |
X                   'quoted string'
X    NB: A quoted string cannot contain embedded single quotes. There
X        is no escape character available in a quoted string.
X
X    rm,   remove files
X         rm [-i] <regular expression>
X         -i ::= interactive mode, you are prompted before
X                file(s) are removed.
X
X    cp,   copy files
X         cp <many files> directory
X              copy all files into directory
X              cp *.c d:\foo - copy all C files into foo
X                           directory on drive D:
X         cp <file> <directory>
X              copy file to directory\file.
X
X         cp <file> <file>
X              file to file copy
X              'cp file con:' to get the listing of a file.
X
X    ls,   list directory
X         ls [wild card]
X         ls without an argument will list all files in the
X         current directory. If an argument is given, then
X         only files matching the wild card will be listed.
X              ls *.c - list all C files in current directory
X              ls d:\foo - list all files in foo directory on D:
X              ls d:\foo\*.c - list all C files in foo directory
X
X    cd,   change working directory
X         cd directory
X
X    md,   make a directory
X         md directory
X
X    rd,   remove a directory
X         rd directory
X         Due to a Gemdos bug, you may have to say 'rd directory'
X         twice to get rid of directory. This happens when you
```

```
X         try to remove a directory that you have never visited.
X
X    pwd,  print  working directory
X         prints the current working directory
X
X    df,   check free space
X         df [device]
X         df without an argument will show free space on current drive.
X         df with an argument will show free space on specified drive.
X
XTERMCAP
X    Termcap entries that may be used for the terminal emulator.
X
Xst|520st|atariST|520 or 1040, bw, std sys font, 25 lines, 80 col:\
X    :ae=\Eba:al=\EL:am:as=\Ebc:\
X    :bl=^G:bs:\
X    :cd=\EJ:ce=\EK:cl=\EH\EJ:cm=\EY%+ %+ :co#80:cr=^M:\
X    :dl=99\EM:do=\EB:\
X    :ho=\EH:\
X    :is=\Ev\Ee:\
X    :kd=\274:kh=\216:kl=\275:kr=\276:ku=\273:\
X    :le=^H:li#25:\
X    :nd=\EC:nl=^J:\
X    :pt:\
X    :se=\Eq:so=\Ep:sr=\EI:\
X    :ta=^I:\
X    :up=\EA:
X
X
XST|ST25|atariSTcolor|as above but with color for standout :\
X    :ae=\Eba:al=\EL:am:as=\Ebc:\
X    :bl=^G:bs:\
X    :cm=\EY%+ %+ :co#80:li#25:cr=^M:cd=\EJ:ce=\EK:cl=\EH\EJ:\
X    :dl=99\EM:do=^J:\
X    :ho=\EH:\
X    :is=\Ev\Ee:\
X    :kd=\274:kh=\216:kl=\275:kr=\276:ku=\273:\
X    :le=^H:\
X    :nd=\EC:nl=^J:\
X    :pt:\
X    :so=\Ec2\Eb3:se=\Ec0\Eb3:sr:\EI:\
X    :ta=^I:\
X    :up=\EA:
X
XsT|st50|AtariST emulating vt52, bw, 50 lines, 80 col:\
X    :ae=\Eba:al=\EL:am:as=\Ebc:\
X    :bl=^G:bs:\
X    :cd=\EJ:ce=\EK:cl=\EH\EJ:cm=\EY%+ %+ :co#80:cr=^M:\
X    :dl=99\EM:do=^J:\
X    :ho=\EH:\
X    :is=\Ev\Ee:\
X    :kd=\274:kh=\216:kl=\275:kr=\276:ku=\273:\
X    :le=^H:li#50:\
```

```
X    :nd=\EC:nl=^J:\
X    :pt:\
X    :se=\Eq:so=\Ep:sr=\EI:\
X    :ta=^I:\
X    :up=\EA:
X
XCOMPILING
X
X    The source provided with this distribution will compile
X    with either ALCYON C V4.14 as distributed with the
X    Atari Development System,  with Mark Williams C Version 2.00
X    or later and with MANX Aztec C (V3.6a tested).
X    (please note that the system will NOT compile with earlier
X     versions of either compiler.)
X
X    Edit config.h.
X
X    If you are using Alcyon(and ALN for the linker), check paths in LNK
X
X    Define the preprocessor symbol `RECURSE' if you
X    want the sz to accept directory names(and send its
X    contents), and for the -P option.
X
X    To compile with Alcyon see the file `makefile.alc'
X    To compile with Mark Williams C see the file `makefile'
X    To compile with Manx Aztec C see the file `makefile.man'
X
X    Rename ZMDM.PRG to ZMDM.TOS if you so desire.
X
XMORE ALCYON NOTES:
X    -- Some people seem to have a bad version of as68 with Alcyon,
X     that does'nt handle static variables too well (ie. symbols of the
X     form "`name" in the assembler). Please make sure yours is not one of
X     these.
X
X    -- Use a reasonable version of gemstart.s - ie. one which gives you
X     a decent amount of stack+heap space. One of the nicer versions
X     floating around is the one written by Alan Pratt @atari, that
X     gives you the various memory model options, and corrects the
X     bug with the bdos call as suggested by Robert Royar on this net.
X
X    -- Do NOT use the `fix' distributed on CompuServe developers forum
X    in the file WILDFI.ARC (in Dl7). This fix does'nt fix anything,
X    it breaks everything!
X
XKNOWN BUGS
X    -v's on the ST end don't do a thing!
X
X    If a receive is cancelled/aborted, the receive file
X    buffer is not flushed to the file. Personally, i
X    view this as a feature rather than a bug, so it is
X    unlikely to be `fixed'.
X
```

```
XDISCLAIMER
X    This code is in public domain, and you are encouraged
X    to distribute it further. You may however not sell the
X    code, or use it for any commercial gains. The code is
X    provided as is, and we are not responsible for any
X    omissions/errors, bugs, nor do we claim correctness or
X    its fitness for any purpose.
X
X    Please forward your comments and suggestions to
X
X                 Jwahar Bammi
X             usenet: mandrill!bammi@{decvax,sun}.UUCP
X             csnet:  bammi@mandrill.ces.CWRU.edu
X             arpa:   bammi@mandrill.ces.CWRU.edu
X             CompuServe: 71515,155
X
X
X/* EOF */
SHAR_EOF
chmod 0600 ZMDM.DOC || echo "restore of ZMDM.DOC fails"
echo "x - extracting ZMDM.H (Text)"
sed 's/^X//' << 'SHAR_EOF' > ZMDM.H &&
X/*
X *  Common include file
X *
X *      Jwahar Bammi
X *           usenet: mandrill!bammi@{decvax,sun}.UUCP
X *           csnet:  bammi@mandrill.ces.CWRU.edu
X *           arpa:   bammi@mandrill.ces.CWRU.edu
X *           CompuServe: 71515,155
X */
X
X#include <stdio.h>
X#include <ctype.h>
X#include <setjmp.h>
X#include <osbind.h>
X#include "decl.h"
X
X    /* See configurable parameters towards the end of the file */
X    /* Leave the rest alone                     */
X
X
X
X    /* Common defines */
X#ifndef FALSE
X#define  FALSE         0
X#define TRUE           1
X#endif
X
X
X#define EscSeq(x) Bconout(2,(int)'\033'); Bconout(2,(int)x) /* Send Esc.
X                                 Seq. to   console */
X#define CTRL(X)       (X & 037) /* CTRL-anything */
```

```
X
X#define STOS   "GEMDOS/TOS"
X#define ZMDMVERSION    "1.62 -- 05/19/88"
X
X/* Ward Christensen / CP/M parameters - Don't change these! */
X#define ENQ 005
X#define CAN ('X'&037)
X#define XOFF ('s'&037)
X#define XON ('q'&037)
X#define SOH 1
X#define STX 2
X#define EOT 4
X#define ACK 6
X#define NAK 025
X#define CPMEOF 032
X#define WANTCRC 0103  /* send C not NAK to get crc not checksum */
X#define TIMEOUT (-2)
X#define RCDO (-3)
X#define ERRORMAX 5
X#define RETRYMAX 10   /* for dogs like compuserve */
X#define WCEOT (-10)
X#define SECSIZ 128    /* cp/m's Magic Number record size */
X#define PATHLEN 128   /* ready for 4.2 bsd ? noooooo */
X#define KSIZE 1024    /* record size with k option */
X#define UNIXFILE 0x8000    /* happens to the the S_IFREG file mask bit for stat
*/
X#define DEFBYTL 2000000000L     /* default rx file size */
X#define WANTG 0107    /* Send G not NAK to get nonstop batch xmsn */
X
X#define PURGELINE while(Bconstat(1))Bconin(1);Lleft = 0
X
X#define Bauxws(X)  wr_modem(X)
X
X#define RLOGFILE "rzlog"
X#define zperr vfile
X#define zperr2 vfile2
X
X#define OK 0
X
X#ifdef ERROR
X#undef ERROR
X#endif
X#define ERROR (-1)
X
X#define HOWMANY 133
X/* Parameters for ZSINIT frame */
X#define ZATTNLEN 32   /* Max length of attention string */
X
X
X    /* Types */
X
X#ifdef LONG
X#undef LONG  /* Get rid of stupid portab.h definition */
```

```
X#endif
X
X#ifdef WORD
X#undef WORD
X#endif
X
X#ifdef UWORD
X#undef UWORD
X#endif
X
Xstruct  stat
X{
X    char st_sp1[21];    /* Junk     */
X    char st_mode;       /* File attributes */
X    int  st_time;       /* Mod Time    */
X    int     st_date;       /* Mod date     */
X    long st_size;       /* File size    */
X    char st_name[14];   /* File name    */
X};
X
X/* The structure returned by Iorec(), really ptr to this type */
Xtypedef struct {
X    char *ibuf;
X    int ibufsiz;
X    int ibufhd;
X    int ibuftl;
X    int ibuflow;
X    int ibufhi;
X} IOREC;
X
Xtypedef  struct {
X    char *sbaud;
X    int  ibaud;
X    int  jbaud;
X} BAUDS;
X
X/*
X * updcrc macro derived from article Copyright (C) 1986 Stephen Satchell.
X *  NOTE: First srgument must be in range 0 to 255.
X *        Second argument is referenced twice.
X *
X * Programmers may incorporate any or all code into their programs,
X * giving proper credit within the source. Publication of the
X * source routines is permitted so long as proper credit is given
X * to Stephen Satchell, Satchell Evaluations and Chuck Forsberg,
X * Omen Technology.
X */
X#ifndef IN_COMMON
Xextern unsigned int crctab[]; /* see definition in common.c */
Xextern unsigned long int crc_32_tab[]; /* see definition in common.c */
X#endif
X
X#define updcrc(cp, crc) \
```

```
X( crctab[(((unsigned int)crc >> 8) & 255)] ^ ((unsigned int)crc << 8) ^ (unsigned
int)cp)
X
X#define UPDC32(octet, crc) \
X(crc_32_tab[(((unsigned long)crc) ^ ((unsigned long)octet)) & 0xff] ^ (((unsigned
long)crc) >> 8))
X
X
X#define BAUD_STRING(s)((s < 0) ? "**UNKNOWN**" : vbauds[s].sbaud)
X#define BAUD_RATE(s)  ((s < 0) ? 9600 : vbauds[s].jbaud)
X
X/* -eof- */
SHAR_EOF
chmod 0600 ZMDM.H || echo "restore of ZMDM.H fails"
echo "x - extracting ZMODEM.H (Text)"
sed 's/^X//' << 'SHAR_EOF' > ZMODEM.H &&
X/*
X *               ACKNOWLEDGEMENTS
X *
X *  ZMDM was derived from rz/sz for Unix  posted by
X *  Chuck Forsberg (...!tektronix!reed!omen!caf ). We
X *  thank him for his excellent code, and for giving
X *  us permission to use and distribute his code and
X *  documentation.
X *
X *  Atari St version by:
X *       Jwahar Bammi
X *            usenet: mandrill!bammi@{decvax,sun}.UUCP
X *            csnet:  bammi@mandrill.ces.CWRU.edu
X *            arpa:   bammi@mandrill.ces.CWRU.edu
X *            CompuServe: 71515,155
X */
X
X/*
X *   Z M O D E M . H     Manifest constants for ZMODEM
X *    application to application file transfer protocol
X *    01-15-87  Chuck Forsberg Omen Technology Inc
X */
X#define ZPAD '*'  /* 052 Padding character begins frames */
X#define ZDLE 030  /* Ctrl-X Zmodem escape - `ala BISYNC DLE */
X#define ZDLEE (ZDLE^0100)  /* Escaped ZDLE as transmitted */
X#define ZBIN 'A'  /* Binary frame indicator */
X#define ZHEX 'B'  /* HEX frame indicator */
X#define ZBIN32 'C'    /* Binary frame with 32 bit FCS */
X
X/* Frame types (see array "frametypes" in zm.c) */
X#define ZRQINIT   0    /* Request receive init */
X#define ZRINIT    1    /* Receive init */
X#define ZSINIT 2  /* Send init sequence (optional) */
X#define ZACK 3        /* ACK to above */
X#define ZFILE 4       /* File name from sender */
X#define ZSKIP 5       /* To sender: skip this file */
X#define ZNAK 6        /* Last packet was garbled */
```

```
X#define ZABORT 7  /* Abort batch transfers */
X#define ZFIN 8        /* Finish session */
X#define ZRPOS 9       /* Resume data trans at this position */
X#define ZDATA 10  /* Data packet(s) follow */
X#define ZEOF 11       /* End of file */
X#define ZFERR 12  /* Fatal Read or Write error Detected */
X#define ZCRC 13       /* Request for file CRC and response */
X#define ZCHALLENGE 14 /* Receiver's Challenge */
X#define ZCOMPL 15 /* Request is complete */
X#define ZCAN 16       /* Other end canned session with CAN*5 */
X#define ZFREECNT 17   /* Request for free bytes on filesystem */
X#define ZCOMMAND 18   /* Command from sending program */
X#define ZSTDERR 19    /* Output to standard error, data follows */
X
X/* ZDLE sequences */
X#define ZCRCE 'h' /* CRC next, frame ends, header packet follows */
X#define ZCRCG 'i' /* CRC next, frame continues nonstop */
X#define ZCRCQ 'j' /* CRC next, frame continues, ZACK expected */
X#define ZCRCW 'k' /* CRC next, ZACK expected, end of frame */
X#define ZRUB0 'l' /* Translate to rubout 0177 */
X#define ZRUB1 'm' /* Translate to rubout 0377 */
X
X/* zdlread return values (internal) */
X/* -1 is general error, -2 is timeout */
X#define GOTOR 0400
X#define GOTCRCE (ZCRCE|GOTOR)   /* ZDLE-ZCRCE received */
X#define GOTCRCG (ZCRCG|GOTOR)   /* ZDLE-ZCRCG received */
X#define GOTCRCQ (ZCRCQ|GOTOR)   /* ZDLE-ZCRCQ received */
X#define GOTCRCW (ZCRCW|GOTOR)   /* ZDLE-ZCRCW received */
X#define GOTCAN    (GOTOR|030)   /* CAN*5 seen */
X
X/* Byte positions within header array */
X#define ZF0  3    /* First flags byte */
X#define ZF1  2
X#define ZF2  1
X#define ZF3  0
X#define ZP0  0    /* Low order 8 bits of position */
X#define ZP1  1
X#define ZP2  2
X#define ZP3  3    /* High order 8 bits of file position */
X
X/* Bit Masks for ZRINIT flags byte ZF0 */
X#define CANFDX    01   /* Rx can send and receive true FDX */
X#define CANOVIO   02   /* Rx can receive data during disk I/O */
X#define CANBRK    04   /* Rx can send a break signal */
X#define CANCRY    010  /* Receiver can decrypt */
X#define CANLZW    020  /* Receiver can uncompress */
X#define CANFC32   040  /* Receiver can use 32 bit Frame Check */
X
X
X/* Parameters for ZFILE frame */
X/* Conversion options one of these in ZF0 */
X#define ZCBIN     1    /* Binary transfer - inhibit conversion */
```

```
X#define ZCNL 2    /* Convert NL to local end of line convention */
X#define ZCRESUM   3    /* Resume interrupted file transfer */
X/* Management options, one of these in ZF1 */
X#define ZMNEW     1    /* Transfer if source newer or longer */
X#define ZMCRC     2    /* Transfer if different file CRC or length */
X#define ZMAPND    3    /* Append contents to existing file (if any) */
X#define ZMCLOB    4    /* Replace existing file */
X#define ZMSPARS   5    /* Encoding for sparse file */
X#define ZMDIFF    6    /* Transfer if dates or lengths different */
X#define ZMPROT    7    /* Protect destination file */
X/* Transport options, one of these in ZF2 */
X#define ZTLZW     1    /* Lempel-Ziv compression */
X#define ZTCRYPT   2    /* Encryption */
X#define ZTRLE     3    /* Run Length encoding */
X
X/* Parameters for ZCOMMAND frame ZF0 (otherwise 0) */
X#define ZCACK1    1    /* Acknowledge, then do command */
X
Xextern long rclhdr();
X
X/* -eof- */
SHAR_EOF
chmod 0600 ZMODEM.H || echo "restore of ZMODEM.H fails"
rm -f s2_seq_.tmp
echo "You have unpacked the last part"
exit 0
```

```
From 1349186664814598221
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,74c8c4f92374eccd,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-06-01 13:51:45 PST
Path: utzoo!utgpu!water!watmath!clyde!att!osu-cis!tut.cis.ohio-state.edu!bloom-
beacon!mit-eddie!ll-xn!ames!pasteur!ucbvax!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i049: ops5 -- OPS5 System in Cambridge Lisp  part01/03
Keywords: shar, lisp
Message-ID: <3561@saturn.ucsc.edu>
Date: 1 Jun 88 20:51:45 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1122
Approved: koreth@ssyx.ucsc.edu

Submitted-by: cfc@wjh12.harvard.edu (Christopher F. Chabris)
Posting-number: Volume 1, Issue 49
Archive-name: ops5/part01

[I don't have Cambridge Lisp, so I can't test this. -sg]

In the past, several people have asked me about a version of the OPS5
production system interpreter for Cambridge Lisp on the ST. This version
```

was given to me by John Fitch, who implemented the ST version of Cambridge Lisp, about two years ago. I submit it with no claims as to workability, usefulness, bug-freeness, etc -- it is just here for those who want to hack with OPS5 and a large program on Cambridge Lisp and the ST. It might also work for other implementations of Cambridge Lisp on other machines, such as the Amiga. And of course, I am not responsible in any way for supporting it. I hope some people enjoy it.

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
# This is part 1 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
#
#    Run the following text with /bin/sh to create:
#      ops5
#
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
sed 's/^X//' << 'SHAR_EOF' > ops5 &&
X%============================================================================
X% OPS5 heavily modified by John Fitch to improve efficiency and
X% functionality.  This is a Cambridge LISP version.
X
X%%% Compatability for Cambridge LISP
X
X(setsyntax ":=+-*$&?<>" 'break!-character nil)
X(setsyntax ":=+-*$&?<>" 'letter t)
X
X(setq !!excise excise)
X(car!-nil!-legal t)
X
X(dm flatc (n) (list 'length (list 'explode n)))
X
X(dm putprop (x y z) (list 'put x z y))   % Argument order
X(dm !!mapc (x y) (list 'mapc y x))  % Argument order
X(dm !!minus (x) (minus x))      % Possible syntax problem
X(dm prog1 l
X    (prog (var)
X      (setq var (gensym))
X      (return
X       `(prog (,var)
X          (setq ,var ,(car l))
X          ,@(append (cdr l) '((return ,xxx)))))))
X
X(setq !*comp t)
X
X% In general the IO model of Cambridge LISP differs from Franz
X% These functions patch it up a little
X
X(de !!read (prt)
```

```
X   (prog (x ans)
X      (setq x (rds prt))
X      (setq ans (read))
X      (rds x)
X      (return ans)))
X
X(de !!tyipeek (prt)
X   (prog (x ans)
X      (setq x (rds prt 'input))
X      (setq ans (tyipeek))
X      (rds x)
X      (return ans)))
X
X(de !!princ (x prt)
X  (prog (old)
X      (setq old (rds prt))
X      (princ x)
X      (rds old)
X      (return x)))
X
X% Useful function not defined in Cambridge LISP
X
X(de delq (a b)
X     (cond
X        ((null b) nil)
X        ((eq a (car b)) (cdr b))
X        (t (cons (car b) (delq a (cdr b)))) ))
X
X(fluid
X   '(*matrix* *buckets* *accept-file* *write-file* *trace-file*
X      *class-list* *brkpts* *strategy* *in-rhs* *ptrace* *wtrace*
X      *recording* *refracts* *real-cnt* *virtual-cnt* *max-cs*
X      *total-cs* *limit-token* *limit-cs* *critical* *build-trace*
X      *wmpart-list* *size-result-array* *result-array*
X      *record-array* *result-array* *size-result-array*
X      *pcount* *cycle-count* *action-count* *total-token* *max-token*
X      *current-token* *total-cs* *max-cs* *total-wm* *max-wm*
X      *current-wm* *conflict-set* *wmpart-list* *p-name*
X      *remaining-cycles* *first-node*      *feature-count* *cur-vars*
X      *ce-count*      *vars*      *ce-vars*      *rhs-bound-vars*
X      *rhs-bound-ce-vars*      *last-branch* *last-node* *subnum*
X      *record* *max-record-index* *record-index* *curcond* *sendtocall* *side*
X      *flag-part* *data-part* *alpha-flag-part* *alpha-data-part* *wm-filter*
X      *wm* *old-wm* *action-type* *data-matched* *last* *variable-memory*
X      *filters* *ppline* *halt-flag* *ce-variable-memory* *rest* *max-index*
X      *next-index* *break-flag* *phase* *cvec* *cved-least*
X))
X
X
X% =ALG returns T if A and B are algebraicly equal.
X(de =alg (a b) (zerop (difference a b)))
X
X(de ce-gelm (x k)
```

```
X   (prog nil
Xloop  (cond ((eq k 1) (return (car x))))
X      (setq k (isub1 k))
X      (setq x (cdr x))
X      (go loop)))
X
X% The loops in gelm were unwound so that fewer calls on DIFFERENCE
X% would be needed. (JPff comment: Yeak)
X(de gelm (x k)
X   (prog (ce sub)
X      (setq ce (iquotient k 10000))
X      (setq sub (idifference k (itimes ce 10000)))
Xceloop(cond ((eq ce 0) (go ph2)))
X      (setq x (cdr x))
X      (cond ((eq ce 1) (go ph2)))
X      (setq x (cdr x))
X      (cond ((eq ce 2) (go ph2)))
X      (setq x (cdr x))
X      (cond ((eq ce 3) (go ph2)))
X      (setq x (cdr x))
X      (cond ((eq ce 4) (go ph2)))
X      (setq ce (idifference ce 4))
X      (go celoop)
Xph2   (setq x (car x))
Xsubloop
X      (cond ((eq sub 0) (go finis)))
X      (setq x (cdr x))
X      (cond ((eq sub 1) (go finis)))
X      (setq x (cdr x))
X      (cond ((eq sub 2) (go finis)))
X      (setq x (cdr x))
X      (cond ((eq sub 3) (go finis)))
X      (setq x (cdr x))
X      (cond ((eq sub 4) (go finis)))
X      (setq x (cdr x))
X      (cond ((eq sub 5) (go finis)))
X      (setq x (cdr x))
X      (cond ((eq sub 6) (go finis)))
X      (setq x (cdr x))
X      (cond ((eq sub 7) (go finis)))
X      (setq x (cdr x))
X      (cond ((eq sub 8) (go finis)))
X      (setq sub (idifference sub 8))
X      (go subloop)
Xfinis (return (car x))))
X
X
X%%% Utility functions
X
X(de printline (x)
X    (foreach y in x do
X    (progn
X        (princ " ")
```

```
X       (print y))))
X
X(de printlinec (x)
X    (foreach y in x do
X    (progn
X        (princ " ")
X        (princ y))))
X
X% intersect two lists using eq for the equality test
X(de interq (x y)
X   (cond
X      ((atom x) nil)
X      ((memq (car x) y) (cons (car x) (interq (cdr x) y)))
X      (t (interq (cdr x) y))))
X
X(de i-g-v nil
X   (prog (x)
X      %(sstatus translink t)
X      %(setsyntax '{ 66)
X      %(setsyntax '} 66)
X      %(setsyntax '^ 66)
X      (setq *buckets* 64)
X      % OPS5 allows 64 named slots
X      (setq *accept-file* nil)
X      (setq *write-file* nil)
X      (setq *trace-file* nil)
X      (setq *class-list* nil)
X      (setq *brkpts* nil)
X      (setq *strategy* 'lex)
X      (setq *in-rhs* nil)
X      (setq *ptrace* t)
X      (setq *wtrace* nil)
X      (setq *recording* nil)
X      (setq *refracts* nil)
X      (setq *real-cnt* (setq *virtual-cnt* 0))
X      (setq *max-cs* (setq *total-cs* 0))
X      (setq *limit-token* 1000000)
X      (setq *limit-cs* 1000000)
X      (setq *critical* nil)
X      (setq *build-trace* nil)
X      (setq *wmpart-list* nil)
X      (setq *size-result-array* 127)
X      (setq *result-array* (mkvect *size-result-array*))
X      (setq *record-array* (mkvect *size-result-array*))
X      %%% Used to be 6 "!!!
X      (setq x 0)
Xloop  (putv *result-array* x nil)
X      (setq x (iadd1 x))
X      (cond ((not (igreaterp x *size-result-array*)) (go loop)))
X      (make-bottom-node)
X      (setq *pcount* 0)
X      (initialize-record)
X      (setq *cycle-count* (setq *action-count* 0))
```

```
X      (setq *total-token*
X     (setq *max-token* (setq *current-token* 0)))
X      (setq *total-cs* (setq *max-cs* 0))
X      (setq *total-wm* (setq *max-wm* (setq *current-wm* 0)))
X      (setq *conflict-set* nil)
X      (setq *wmpart-list* nil)
X      (setq *p-name* nil)
X      (setq *remaining-cycles* 1000000)
X      (setq *cvec* (mkvect 64))
X      (setq *cvec-least* 0)))
X
X% if the size of result-array changes, change the line in i-g-v which
X% sets the value of *size-result-array*
X(de !%warn (what where)
X   (prog nil
X      (terpri)
X      (princ '!?)
X      (and *p-name* (princ *p-name*))
X      (princ "..")
X      (princ where)
X      (princ "..")
X      (princ what)
X      (return where)))
X
X(de !%error (what where)
X   (!%warn what where)
X   (throw !!error!! '!!error!!))
X
X(de round (x) (fix (plus 0.5 x)))
X
X(de top-levels-eq (la lb)
X   (prog nil
Xlx    (cond
X     ((eq la lb) (return t))
X     ((null la) (return nil))
X     ((null lb) (return nil))
X     ((not (eq (car la) (car lb))) (return nil)))
X      (setq la (cdr la))
X      (setq lb (cdr lb))
X      (go lx)))
X
X
X%%% LITERAL and LITERALIZE
X
X(df literal z
X   (prog (atm val old)
Xtop   (cond
X     ((atom z) (return 'bound))
X     ((not (eq (cadr z) '=)) (return (!%warn "wrong format" z))))
X      (setq atm (car z))
X      (setq val (caddr z))
X      (setq z (cdddr z))
X      (cond
```

```
X     ((not (numberp val))
X        (!%warn "can bind only to numbers" val))
X     ((or (not (idp atm)) (variablep atm))
X        (!%warn "can bind only constant atoms" atm))
X     ((and
X         (setq old (literal-binding-of atm))
X         (not (equal old val)))
X        (!%warn "attempt to rebind attribute" atm))
X     (t (put atm 'ops-bind val)))
X      (go top)))
X
X(dm have-compiled-production nil '(not (izerop *pcount*)))
X
X(df literalize l
X   (prog (class-name atts)
X      (setq class-name (car l))
X      (cond
X     ((have-compiled-production)
X        (!%warn "literalize called after p" class-name)
X        (return nil))
X     ((get class-name 'att-list)
X        (!%warn "attempt to redefine class" class-name)
X        (return nil)))
X      (setq *class-list* (cons class-name *class-list*))
X      (setq atts (remove-duplicates (cdr l)))
X      (test-attribute-names atts)
X      (mark-conflicts atts atts)
X      (put class-name 'att-list atts)))
X
X(df vector-attribute l
X   (cond
X      ((have-compiled-production)
X     (!%warn "vector-attribute called after p" l))
X      (t (test-attribute-names l)
X     (flag l 'vector-attribute))))
X
X(dm is-vector-attribute (att) `(flagp ,att 'vector-attribute))
X
X(de test-attribute-names (l)
X   (!!mapc (function test-attribute-names2) l))
X
X(de test-attribute-names2 (atm)
X   (cond
X      ((or (not (idp atm)) (variablep atm))
X     (!%warn "can bind only constant atoms" atm))))
X
X(de finish-literalize nil
X   (cond
X      ((not (null *class-list*))
X     (!!mapc (function note-user-assigns) *class-list*)
X     (!!mapc (function assign-scalars) *class-list*)
X     (!!mapc (function assign-vectors) *class-list*)
X     (!!mapc (function put-ppdat) *class-list*)
```

```
X     (!!mapc (function erase-literal-info) *class-list*)
X     (setq *class-list* nil)
X     (setq *buckets* nil))))
X
X(de put-ppdat (class)
X   (prog (al att ppdat)
X      (setq ppdat nil)
X      (setq al (get class 'att-list))
Xtop   (cond
X     ((not (atom al))
X        (setq att (car al))
X        (setq al (cdr al))
X        (setq ppdat
X           (cons (cons (literal-binding-of att) att) ppdat))
X        (go top)))
X      (putprop class ppdat 'ppdat)))
X
X
X% note-user-assigns and note-user-vector-assigns are needed only when
X% literal and literalize are both used in a program.  They make sure that
X% the assignments that are made explicitly with literal do not cause problems
X% for the literalized classes.
X(de note-user-assigns (class)
X   (!!mapc (function note-user-assigns2) (get class 'att-list)))
X
X(de note-user-assigns2 (att)
X   (prog (num conf buck clash)
X      (setq num (literal-binding-of att))
X      (cond ((null num) (return nil)))
X      (setq conf (get att 'conflicts))
X      (setq buck (store-binding att num))
X      (setq clash (find-common-atom buck conf))
X      (and
X     clash
X     (!%warn
X        "attributes in a class assigned the same number"
X        (cons att clash)))
X      (return nil)))
X
X(de note-user-vector-assigns (att given needed)
X   (and
X      (greaterp needed given)
X      (!%warn
X     "vector attribute assigned too small a value in literal"
X     att)))
X
X(de assign-scalars (class)
X   (!!mapc (function assign-scalars2) (get class 'att-list)))
X
X(de assign-scalars2 (att)
X   (prog (tlist num bucket conf)
X      (cond
X     ((literal-binding-of att) (return nil))
```

```
X     ((is-vector-attribute att) (return nil)))
X      (setq tlist (buckets))
X      (setq conf (get att 'conflicts))
Xtop   (cond
X     ((atom tlist)
X        (!%warn "could not generate a binding" att)
X        (store-binding att (!!minus 1))
X        (return nil)))
X      (setq num (caar tlist))
X      (setq bucket (cdar tlist))
X      (setq tlist (cdr tlist))
X      (cond
X     ((disjoint bucket conf) (store-binding att num))
X     (t (go top)))) )
X
X(de assign-vectors (class)
X   (!!mapc (function assign-vectors2) (get class 'att-list)))
X
X(de assign-vectors2 (att)
X   (prog (big conf new old need)
X      (cond ((not (is-vector-attribute att)) (return nil)))
X      (setq big 1)
X      (setq conf (get att 'conflicts))
Xtop   (cond
X     ((not (atom conf))
X        (setq new (car conf))
X        (setq conf (cdr conf))
X        (cond
X           ((is-vector-attribute new)
X           (!%warn
X              "class has two vector attributes"
X              (list att new)))
X           (t (setq big (max (literal-binding-of new) big))))
X        (go top)))
X      (setq need (iadd1 big))
X      (setq old (literal-binding-of att))
X      (cond
X     (old (note-user-vector-assigns att old need))
X     (t (store-binding att need)))
X      (return nil)))
X
X(de disjoint (la lb) (not (find-common-atom la lb)))
X
X(de find-common-atom (la lb)
X   (prog nil
Xtop   (cond
X     ((null la) (return nil))
X     ((memq (car la) lb) (return (car la)))
X     (t (setq la (cdr la)) (go top)))) )
X
X(de mark-conflicts (rem all)
X   (cond
X      ((not (null rem))
```

```
X     (mark-conflicts2 (car rem) all)
X     (mark-conflicts (cdr rem) all))))
X
X(de mark-conflicts2 (atm lst)
X   (prog (l)
X      (setq l lst)
Xtop   (cond ((atom l) (return nil)))
X      (conflict atm (car l))
X      (setq l (cdr l))
X      (go top)))
X
X(de conflict (a b)
X   (prog (old)
X      (setq old (get a 'conflicts))
X      (and
X     (not (eq a b))
X     (not (memq b old))
X     (putprop a (cons b old) 'conflicts))))
X
X(de remove-duplicates (lst)
X   (cond
X      ((atom lst) nil)
X      ((memq (car lst) (cdr lst)) (remove-duplicates (cdr lst)))
X      (t (cons (car lst) (remove-duplicates (cdr lst)))) ))
X
X(de literal-binding-of (name) (get name 'ops-bind))
X
X(de store-binding (name lit)
X   (putprop name lit 'ops-bind)
X   (add-bucket name lit))
X
X(de add-bucket (name num)
X   (prog (buc)
X      (setq buc (assoc num (buckets)))
X      (and (not (memq name buc)) (rplacd buc (cons name (cdr buc))))
X      (return buc)))
X
X(de buckets nil
X   (and (atom *buckets*) (setq *buckets* (make-nums *buckets*)))
X   *buckets*)
X
X(de make-nums (k)
X   (prog (nums)
X      (setq nums nil)
Xl     (cond ((ilessp k 2) (return nums)))
X      (setq nums (cons (ncons k) nums))
X      (setq k (isub1 k))
X      (go l)))
X
X(de erase-literal-info (class)
X   (!!mapc (function erase-literal-info2) (get class 'att-list))
X   (remprop class 'att-list))
X
```

```
X(de erase-literal-info2 (att) (remprop att 'conflicts))
X
X%%% LHS Compiler
X
X(df p z
X   (finish-literalize)
X   (princ '*)
X   (compile-production (car z) (cdr z))
X   (car z))
X
X(de compile-production (name matrix)
X   (prog (erm)
X      (setq *p-name* name)
X      (setq erm (catch !!error!! (cmp-p name matrix)))
X      (setq *p-name* nil)))
X
X(de peek-lex nil (car *matrix*))
X
X(de lex nil
X   (prog1 (car *matrix*) (setq *matrix* (cdr *matrix*))))
X
X(de end-of-p nil (atom *matrix*))
X
X(de rest-of-p nil *matrix*)
X
X(de prepare-lex (prod) (setq *matrix* prod))
X
X(de peek-sublex nil (car *curcond*))
X
X(de sublex nil
X   (prog1 (car *curcond*) (setq *curcond* (cdr *curcond*))))
X
X(de end-of-ce nil (atom *curcond*))
X
X(de rest-of-ce nil *curcond*)
X
X(de prepare-sublex (ce) (setq *curcond* ce))
X
X(de make-bottom-node nil (setq *first-node* (list '&bus nil)))
X
X(de cmp-p (name matrix)
X   (prog (m bakptrs)
X      (cond
X     ((or (null name) (pairp name))
X        (!%error "illegal production name" name))
X     ((equal (get name 'production) matrix) (return nil)))
X      (prepare-lex matrix)
X      (excise-p name)
X      (setq bakptrs nil)
X      (setq *pcount* (iadd1 *pcount*))
X      (setq *feature-count* 0)
X      (setq *ce-count* 0)
X      (setq *vars* nil)
```

```
X      (setq *ce-vars* nil)
X      (setq *rhs-bound-vars* nil)
X      (setq *rhs-bound-ce-vars* nil)
X      (setq *last-branch* nil)
X      (setq m (rest-of-p))
Xl1    (and (end-of-p) (!%error "no '-->' in production" m))
X      (cmp-prin)
X      (setq bakptrs (cons *last-branch* bakptrs))
X      (cond ((not (eq '--> (peek-lex))) (go l1)))
X      (lex)
X      (check-rhs (rest-of-p))
X      (link-new-node
X     (list '&p *feature-count* name (encode-dope)
X        (encode-ce-dope) (cons 'progn (rest-of-p))))
X      (putprop name (cdr (reversewoc bakptrs)) 'backpointers)
X      (putprop name matrix 'production)
X      (putprop name *last-node* 'topnode)))
X
X(de rating-part (pnode) (cadr pnode))
X
X(de var-part (pnode) (car (cdddr pnode)))
X
X(de ce-var-part (pnode) (cadr (cdddr pnode)))
X
X(de rhs-part (pnode) (caddr (cdddr pnode)))
X
X(de excise-p (name)
X   (cond
X      ((get name 'topnode)
X     (printline (list name 'is 'excised))
X     (setq *pcount* (isub1 *pcount*))
X     (remove-from-conflict-set name)
X     (kill-node (get name 'topnode))
X     (remprop name 'production)
X     (remprop name 'backpointers)
X     (remprop name 'topnode))))
X
X(de kill-node (node)
X   (prog nil
Xtop   (cond ((atom node) (return nil)))
X      (rplaca node '&old)
X      (setq node (cdr node))
X      (go top)))
X
X(de cmp-prin nil
X   (prog nil
X      (setq *last-node* *first-node*)
X      (cond
X     ((null *last-branch*) (cmp-posce) (cmp-nobeta))
X     ((eq (peek-lex) '-) (cmp-negce) (cmp-not))
X     (t (cmp-posce) (cmp-and))) )
X
X(de cmp-negce nil (lex) (cmp-ce))
```

```
X
X(de cmp-posce nil
X   (setq *ce-count* (iadd1 *ce-count*))
X   (cond ((eq (peek-lex) '!{) (cmp-ce+cevar)) (t (cmp-ce))))
X
X(de cmp-ce+cevar nil
X   (prog (z)
X      (lex)
X      (cond
X     ((atom (peek-lex)) (cmp-cevar) (cmp-ce))
X     (t (cmp-ce) (cmp-cevar)))
X      (setq z (lex))
X      (or (eq z '!}) (!%error "missing '}" z))))
X
X(de new-subnum (k)
X   (or (numberp k) (!%error "tab must be a number" k))
X   (setq *subnum* (fix k)))
X
X(de incr-subnum nil (setq *subnum* (iadd1 *subnum*)))
X
X(de cmp-ce nil
X   (prog (z)
X      (new-subnum 0)
X      (setq *cur-vars* nil)
X      (setq z (lex))
X      (and (atom z) (!%error "atomic conditions are not allowed" z))
X      (prepare-sublex z)
Xla    (cond ((end-of-ce) (return nil)))
X      (incr-subnum)
X      (cmp-element)
X      (go la)))
X
X(de cmp-element nil
X   (and (eq (peek-sublex) '!^) (cmp-tab))
X   (cond
X      ((eq (peek-sublex) '!{) (cmp-product))
X      (t (cmp-atomic-or-any))))
X
X(de cmp-atomic-or-any nil
X   (cond ((eq (peek-sublex) '<<) (cmp-any)) (t (cmp-atomic))))
X
X(de cmp-any nil
X   (prog (a z)
X      (sublex)
X      (setq z nil)
Xla    (cond ((end-of-ce) (!%error "missing '>>" a)))
X      (setq a (sublex))
X      (cond ((not (eq '>> a)) (setq z (cons a z)) (go la)))
X      (link-new-node (list '&any nil (current-field) z))))
X
X(de cmp-tab nil
X   (prog (r)
X      (sublex)
```

```
X      (setq r (sublex))
X      (setq r ($litbind r))
X      (new-subnum r)))
X
X(de $litbind (x)
X   (prog (r)
X      (cond
X     ((and (idp x) (setq r (literal-binding-of x)))
X        (return r))
X     (t (return x)))) )
X
X(de get-bind (x)
X   (prog (r)
X      (cond
X     ((and (idp x) (setq r (literal-binding-of x)))
X        (return r))
X     (t (return nil)))) )
X
X(de cmp-atomic nil
X   (prog (test x)
X      (setq x (peek-sublex))
X      (cond
X     ((eq x '=  ) (setq test 'eq) (sublex))
X     ((eq x '<> ) (setq test 'ne) (sublex))
X     ((eq x '<  ) (setq test 'lt) (sublex))
X     ((eq x '<= ) (setq test 'le) (sublex))
X     ((eq x '>  ) (setq test 'gt) (sublex))
X     ((eq x '>= ) (setq test 'ge) (sublex))
X     ((eq x '<=>) (setq test 'xx) (sublex))
X     (t (setq test 'eq)))
X      (cmp-symbol test)))
X
X(de cmp-product nil
X   (prog (save)
X      (setq save (rest-of-ce))
X      (sublex)
Xla    (cond
X     ((end-of-ce)
X        (cond
X           ((member '!} save)
X           (!%error "wrong contex for '}" save))
X           (t (!%error "missing '}" save))))
X     ((eq (peek-sublex) '!}) (sublex) (return nil)))
X      (cmp-atomic-or-any)
X      (go la)))
X
X(de variablep (x)
X    (cond ((not (idp x)) nil)
X      ((flagp x 'nonvariable) nil)
X      ((flagp x 'variable) t)
X      ((eq (car (explode x)) '<)
X       (flag (list x) 'variable)
X       t)
```

```
X      (t (flag (list x) 'nonvariable) nil)))
X
X(de cmp-symbol (test)
X   (prog (flag)
X      (setq flag t)
X      (cond ((eq (peek-sublex) '!/) (sublex) (setq flag nil)))
X      (cond
X     ((and flag (variablep (peek-sublex))) (cmp-var test))
X     ((numberp (peek-sublex)) (cmp-number test))
X     ((idp (peek-sublex)) (cmp-constant test))
X     (t (!%error "unrecognized symbol" (sublex)))) ))
X
X(de cmp-constant (test)
X   (or
X      (memq test '(eq ne xx))
X      (!%error
X     "non-numeric constant after numeric predicate"
X     (sublex)))
X   (link-new-node
X      (list (get test 'ta) nil (current-field) (sublex))))
X
X(de cmp-number (test)
X   (link-new-node
X      (list (get test 'tn) nil (current-field) (sublex))))
X
X(de current-field nil (field-name *subnum*))
X
X(de field-name (num)
X   (cond
X      ((igreaterp num 64)  (!%error "condition is too long" (rest-of-ce)))
X      (t num)))
X
X%%% Compiling variables
X%
X%
X%
X% *cur-vars* are the variables in the condition element currently
X% being compiled.  *vars* are the variables in the earlier condition
X% elements.  *ce-vars* are the condition element variables.  note
X% that the interpreter will not confuse condition element and regular
X% variables even if they have the same name.
X%
X% *cur-vars* is a list of triples: (name predicate subelement-number)
X% eg:           ( (<x> eq 3)
X%                 (<y> ne 1)
X%                 . . . )
X%
X% *vars* is a list of triples: (name ce-number subelement-number)
X% eg:           ( (<x> 3 3)
X%                 (<y> 1 1)
X%                 . . . )
X%
X% *ce-vars* is a list of pairs: (name ce-number)
```

```
X% eg:           ( (ce1 1)
X%                 (<c3> 3)
X%                 . . . )
X(de var-dope (var) (atsoc var *vars*))
X
X(de ce-var-dope (var) (atsoc var *ce-vars*))
X
X(de cmp-var (test)
X   (prog (old name)
X      (setq name (sublex))
X      (setq old (atsoc name *cur-vars*))
X      (cond
X     ((and old (eq (cadr old) 'eq)) (cmp-old-eq-var test old))
X     ((and old (eq test 'eq)) (cmp-new-eq-var name old))
X     (t (cmp-new-var name test)))) )
X
X(de cmp-new-var (name test)
X   (setq *cur-vars* (cons (list name test *subnum*) *cur-vars*)))
X
X(de cmp-old-eq-var (test old)
X   (link-new-node
X      (list
X     (get test 'ts)
X     nil
X     (current-field)
X     (field-name (caddr old)))) )
X
X(de cmp-new-eq-var (name old)
X   (prog (pred next)
X      (setq *cur-vars* (delq old *cur-vars*))
X      (setq next (atsoc name *cur-vars*))
X      (cond
X     (next (cmp-new-eq-var name next))
X     (t (cmp-new-var name 'eq)))
X      (setq pred (cadr old))
X      (link-new-node
X     (list
X        (get pred 'ts)
X        nil
X        (field-name (caddr old))
X        (current-field)))) )
X
X(de cmp-cevar nil
X   (prog (name old)
X      (setq name (lex))
X      (setq old (atsoc name *ce-vars*))
X      (and
X     old
X     (!%error "condition element variable used twice" name))
X      (setq *ce-vars* (cons (list name 0) *ce-vars*))))
X
X(de cmp-not nil (cmp-beta '&not))
X
```

```
X(de cmp-nobeta nil (cmp-beta nil))
X
X(de cmp-and nil (cmp-beta '&and))
X
X(de cmp-beta (kind)
X   (prog (tlist vdope vname vpred vpos old)
X      (setq tlist nil)
Xla    (cond ((atom *cur-vars*) (go lb)))
X      (setq vdope (car *cur-vars*))
X      (setq *cur-vars* (cdr *cur-vars*))
X      (setq vname (car vdope))
X      (setq vpred (cadr vdope))
X      (setq vpos (caddr vdope))
X      (setq old (atsoc vname *vars*))
X      (cond
X     (old (setq tlist (add-test tlist vdope old)))
X     ((neq kind '&not) (promote-var vdope)))
X      (go la)
Xlb    (and kind (build-beta kind tlist))
X      (or (eq kind '&not) (fudge))
X      (setq *last-branch* *last-node*)))
X
X(de add-test (list new old)
X   (prog (ttype lloc rloc)
X      (setq *feature-count* (iadd1 *feature-count*))
X      (setq ttype (get (cadr new) 'tb))
X      (setq rloc (encode-singleton (caddr new)))
X      (setq lloc (encode-pair (cadr old) (caddr old)))
X      (return (cons ttype (cons lloc (cons rloc list)))) ))
X
X% the following two functions encode indices so that gelm can
X% decode them as fast as possible
X(de encode-pair (a b) (iplus (times 10000 (sub1 a)) (isub1 b)))
X
X(de encode-singleton (a) (isub1 a))
X
X(de promote-var (dope)
X   (prog (vname vpred vpos new)
X      (setq vname (car dope))
X      (setq vpred (cadr dope))
X      (setq vpos (caddr dope))
X      (or
X     (eq 'eq vpred)
X     (!%error
X        "illegal predicate for first occurrence"
X        (list vname vpred)))
X      (setq new (list vname 0 vpos))
X      (setq *vars* (cons new *vars*))))
X
X(de fudge nil
X   (!!mapc (function fudge*) *vars*)
X   (!!mapc (function fudge*) *ce-vars*))
X
```

```
X(de fudge* (z)
X   (prog (a)
X      (setq a (cdr z))
X      (rplaca a (iadd1 (car a))) )
X
X(de build-beta (type tests)
X   (prog (rpred lpred lnode lef)
X      (link-new-node (list '&mem nil nil (protomem)))
X      (setq rpred *last-node*)
X      (cond
X     ((eq type '&and)
X        (setq lnode (list '&mem nil nil (protomem))))
X     (t (setq lnode (list '&two nil nil))))
X      (setq lpred (link-to-branch lnode))
X      (cond
X     ((eq type '&and) (setq lef lpred))
X     (t (setq lef (protomem))))
X      (link-new-beta-node (list type nil lef rpred tests))))
X
X(de protomem nil (list nil))
X
X(de memory-part (mem-node) (car (cadddr mem-node)))
X
X(de encode-dope nil
X   (prog (r all z k)
X      (setq r nil)
X      (setq all *vars*)
Xla    (cond ((atom all) (return r)))
X      (setq z (car all))
X      (setq all (cdr all))
X      (setq k (encode-pair (cadr z) (caddr z)))
X      (setq r (cons (car z) (cons k r)))
X      (go la)))
X
X(de encode-ce-dope nil
X   (prog (r all z k)
X      (setq r nil)
X      (setq all *ce-vars*)
Xla    (cond ((atom all) (return r)))
X      (setq z (car all))
X      (setq all (cdr all))
X      (setq k (cadr z))
X      (setq r (cons (car z) (cons k r)))
X      (go la)))
X
X%%% Linking the nodes
X
X(de link-new-node (r)
X   (cond
X      ((not (member (car r) '(&p &mem &two &and &not)))
X     (setq *feature-count* (iadd1 *feature-count*))))
X   (setq *virtual-cnt* (iadd1 *virtual-cnt*))
X   (setq *last-node* (link-left *last-node* r)))
```

```
X
X(de link-to-branch (r)
X   (setq *virtual-cnt* (iadd1 *virtual-cnt*))
X   (setq *last-branch* (link-left *last-branch* r)))
X
X(de link-new-beta-node (r)
X   (setq *virtual-cnt* (iadd1 *virtual-cnt*))
X   (setq *last-node* (link-both *last-branch* *last-node* r))
X   (setq *last-branch* *last-node*))
X
X(de link-left (pred succ)
X   (prog (a r)
X      (setq a (left-outs pred))
X      (setq r (find-equiv-node succ a))
X      (cond (r (return r)))
X      (setq *real-cnt* (iadd1 *real-cnt*))
X      (attach-left pred succ)
X      (return succ)))
X
X(de link-both (left right succ)
X   (prog (a r)
X      (setq a (interq (left-outs left) (right-outs right)))
X      (setq r (find-equiv-beta-node succ a))
X      (cond (r (return r)))
X      (setq *real-cnt* (iadd1 *real-cnt*))
X      (attach-left left succ)
X      (attach-right right succ)
X      (return succ)))
X
X(de attach-right (old new)
X   (rplaca (cddr old) (cons new (caddr old))))
X
X(de attach-left (old new) (rplaca (cdr old) (cons new (cadr old))))
X
X(de right-outs (node) (caddr node))
X
X(de left-outs (node) (cadr node))
X
X(de find-equiv-node (node list)
X   (prog (a)
X      (setq a list)
Xl1    (cond
X     ((atom a) (return nil))
X     ((equiv node (car a)) (return (car a))))
X      (setq a (cdr a))
X      (go l1)))
X
X(de find-equiv-beta-node (node list)
X   (prog (a)
X      (setq a list)
Xl1    (cond
X     ((atom a) (return nil))
X     ((beta-equiv node (car a)) (return (car a))))
```

```
X      (setq a (cdr a))
X      (go l1)))
X
X% do not look at the predecessor fields of beta nodes; they have to be
X% identical because of the way the candidate nodes were found
X(de equiv (a b)
X   (and
X      (eq (car a) (car b))
X      (or
X     (eq (car a) '&mem)
X     (eq (car a) '&two)
X     (equal (caddr a) (caddr b)))
X      (equal (cdddr a) (cdddr b))))
X
X(de beta-equiv (a b)
X   (and
X      (eq (car a) (car b))
X      (equal (cddddr a) (cddddr b))
X      (or (eq (car a) '&and) (equal (caddr a) (caddr b)))) )
X
X% the equivalence tests are set up to consider the contents of
X% node memories, so they are ready for the build action
X
X
X%%% Network interpreter
X
X(de match (flag wme)
X   (sendto flag (list wme) 'left (list *first-node*)))
X
X% note that eval-nodelist is not set up to handle building
X% productions.  would have to add something like ops4's build-flag
X(de eval-nodelist (nl)
X   (prog nil
Xtop   (cond ((null nl) (return nil)))
X      (setq *sendtocall* nil)
X      (setq *last-node* (car nl))
X      (apply (caar nl) (cdar nl))
X      (setq nl (cdr nl))
X      (go top)))
X
X(de sendto (flag data side nl)
X   (prog nil
Xtop   (cond ((not nl) (return nil)))
X      (setq *side* side)
X      (setq *flag-part* flag)
X      (setq *data-part* data)
X      (setq *sendtocall* t)
X      (setq *last-node* (car nl))
X      (apply (caar nl) (cdar nl))
X      (setq nl (cdr nl))
X      (go top)))
X
X% &bus sets up the registers for the one-input nodes.
```

```
X% Heavily modified by JPff
X
X(de &bus (outs)
X   (prog (dp i)
X      (setq i 1)
X      (setq *alpha-flag-part* *flag-part*)
X      (setq dp (car (setq *alpha-data-part* *data-part*)))
X      (while dp
X    (progn
X         (putv *cvec* i (car dp))
X         (setq i (iadd1 i))
X         (setq dp (cdr dp))))
X      (setq i (isub1 i))
X      (cond
X     ((ilessp i *cvec-least*)
X      (prog (j)
X        (setq j (iadd1 i))
X        (while (ilessp j *cvec-least*)
X         (progn (putv *cvec* j nil)
X                (setq j (iadd1 j)))))))
X      (setq *cvec-least* i)
X      (eval-nodelist outs)))
X
X(de &any (outs register const-list)
X   (prog (z c)
X      (setq z (getv *cvec* register))
X      (cond ((numberp z) (go number)))
Xsymbol(cond
X     ((null const-list) (return nil))
X     ((eq (car const-list) z) (go ok))
X     (t (setq const-list (cdr const-list)) (go symbol)))
Xnumber(cond
X     ((null const-list) (return nil))
X     ((and (numberp (setq c (car const-list))) (=alg c z))
X        (go ok))
X     (t (setq const-list (cdr const-list)) (go number)))
Xok    (eval-nodelist outs)))
X
X(de teqa (outs register constant)
X   (and (eq (getv *cvec* register) constant) (eval-nodelist outs)))
X
X(put 'eq 'ta 'teqa)
X
X(de tnea (outs register constant)
X   (and
X      (not (eq (getv *cvec* register) constant))
X      (eval-nodelist outs)))
X
X(put 'ne 'ta 'tnea)
X
X(de txxa (outs register constant)
X   (and (idp (getv *cvec* register)) (eval-nodelist outs)))
X
```

```
X(put 'xx 'ta 'txxa)
X
X(de teqn (outs register constant)
X   (prog (z)
X      (setq z (getv *cvec* register))
X      (and (numberp z) (=alg z constant) (eval-nodelist outs))))
X
X(put 'eq 'tn 'teqn)
X
X(de tnen (outs register constant)
X   (prog (z)
X      (setq z (getv *cvec* register))
X      (and
X     (or (not (numberp z)) (not (=alg z constant)))
X     (eval-nodelist outs))))
X
X(put 'ne 'tn 'tnen)
X
X(de txxn (outs register constant)
X    (and (numberp (getv *cvec* register)) (eval-nodelist outs)))
X
X(put 'xx 'tn 'txxn)
X
X(de tltn (outs register constant)
X   (prog (z)
X      (setq z (getv *cvec* register))
X      (and (numberp z) (greaterp constant z) (eval-nodelist outs))))
X
X(put 'lt 'tn 'tltn)
X
X(de tgtn (outs register constant)
X   (prog (z)
X      (setq z (getv *cvec* register))
X      (and (numberp z) (greaterp z constant) (eval-nodelist outs))))
X
X(put 'gt 'tn 'tgtn)
X
X(de tgen (outs register constant)
X   (prog (z)
X      (setq z (getv *cvec* register))
X      (and
X     (numberp z)
X     (not (greaterp constant z))
X     (eval-nodelist outs))))
X
X(put 'ge 'tn 'tgen)
X
X(de tlen (outs register constant)
X   (prog (z)
X      (setq z (getv *cvec* register))
SHAR_EOF
echo "End of part 1, continue with part 2"
echo "2" > s2_seq_.tmp
```

```
exit 0


From 2386449906598885979
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,f5140d8534b90d23,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-06-01 13:52:28 PST
Path: utzoo!utgpu!water!watmath!clyde!att!osu-cis!tut.cis.ohio-state.edu!bloom-
beacon!mit-eddie!ll-xn!ames!pasteur!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i050: ops5 -- OPS5 System in Cambridge Lisp  part02/03
Keywords: shar, lisp
Message-ID: <3562@saturn.ucsc.edu>
Date: 1 Jun 88 20:52:28 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1137
Approved: koreth@ssyx.ucsc.edu

Submitted-by: cfc@wjh12.harvard.edu (Christopher F. Chabris)
Posting-number: Volume 1, Issue 50
Archive-name: ops5/part02

#!/bin/sh
# this is part 2 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file ops5 continued
#
CurArch=2
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> ops5
X      (and
X     (numberp z)
X     (not (greaterp z constant))
X     (eval-nodelist outs))))
X
X(put 'le 'tn 'tlen)
X
X(de teqs (outs vara varb)
X   (prog (a b)
X      (setq a (getv *cvec* vara))
X      (setq b (getv *cvec* varb))
X      (cond
X     ((eq a b) (eval-nodelist outs))
```

```
X     ((and (numberp a) (numberp b) (=alg a b))
X        (eval-nodelist outs)))) )
X
X(put 'eq 'ts 'teqs)
X
X(de tnes (outs vara varb)
X   (prog (a b)
X      (setq a (getv *cvec* vara))
X      (setq b (getv *cvec* varb))
X      (cond
X     ((eq a b) (return nil))
X     ((and (numberp a) (numberp b) (=alg a b)) (return nil))
X     (t (eval-nodelist outs)))) )
X
X(put 'ne 'ts 'tnes)
X
X(de txxs (outs vara varb)
X   (prog (a b)
X      (setq a (getv *cvec* vara))
X      (setq b (getv *cvec* varb))
X      (cond
X     ((and (numberp a) (numberp b)) (eval-nodelist outs))
X     ((and (not (numberp a)) (not (numberp b)))
X        (eval-nodelist outs)))) )
X
X(put 'xx 'ts 'txxs)
X
X(de tlts (outs vara varb)
X   (prog (a b)
X      (setq a (getv *cvec* vara))
X      (setq b (getv *cvec* varb))
X      (and
X     (numberp a)
X     (numberp b)
X     (greaterp b a)
X     (eval-nodelist outs))))
X
X(put 'lt 'ts 'tlts)
X
X(de tgts (outs vara varb)
X   (prog (a b)
X      (setq a (getv *cvec* vara))
X      (setq b (getv *cvec* varb))
X      (and
X     (numberp a)
X     (numberp b)
X     (greaterp a b)
X     (eval-nodelist outs))))
X
X(put 'gt 'ts 'tgts)
X
X(de tges (outs vara varb)
X   (prog (a b)
```

```
X      (setq a (getv *cvec* vara))
X      (setq b (getv *cvec* varb))
X      (and
X     (numberp a)
X     (numberp b)
X     (not (greaterp b a))
X     (eval-nodelist outs))))
X
X(put 'ge 'ts 'tges)
X
X(de tles (outs vara varb)
X   (prog (a b)
X      (setq a (getv *cvec* vara))
X      (setq b (getv *cvec* varb))
X      (and
X     (numberp a)
X     (numberp b)
X     (not (greaterp a b))
X     (eval-nodelist outs))))
X
X(put 'le 'ts 'tles)
X
X(de &two (left-outs right-outs)
X   (prog (fp dp)
X      (cond
X     (*sendtocall* (setq fp *flag-part*) (setq dp *data-part*))
X     (t (setq fp *alpha-flag-part*) (setq dp *alpha-data-part*)))
X      (sendto fp dp 'left left-outs)
X      (sendto fp dp 'right right-outs)))
X
X(de &mem (left-outs right-outs memory-list)
X   (prog (fp dp)
X      (cond
X     (*sendtocall* (setq fp *flag-part*) (setq dp *data-part*))
X     (t (setq fp *alpha-flag-part*) (setq dp *alpha-data-part*)))
X      (sendto fp dp 'left left-outs)
X      (add-token memory-list fp dp nil)
X      (sendto fp dp 'right right-outs)))
X
X(de &and (outs lpred rpred tests)
X    (prog (mem)
X    (cond
X       ((eq *side* 'right)
X        (cond
X         ((not (setq mem (memory-part lpred))) (return nil))
X         (t (and-right outs mem tests))))
X       ((not (setq mem (memory-part rpred))) (return nil))
X       (t (and-left outs mem tests)))))
X
X(de and-left (outs mem tests)
X   (prog (fp dp memdp tlist tst lind rind res)
X      (setq fp *flag-part*)
X      (setq dp *data-part*)
```

```
Xfail  (cond ((null mem) (return nil)))
X      (setq memdp (car mem))
X      (setq mem (cdr mem))
X      (setq tlist tests)
Xtloop (cond ((null tlist) (go succ)))
X      (setq tst (car tlist))
X      (setq tlist (cdr tlist))
X      (setq lind (car tlist))
X      (setq tlist (cdr tlist))
X      (setq rind (car tlist))
X      (setq tlist (cdr tlist))
X      %% the next line differs in and-left & -right
X      (setq res (tst (gelm memdp rind) (gelm dp lind)))
X      (cond (res (go tloop)) (t (go fail)))
Xsucc  %% the next line differs in and-left & -right
X      (sendto fp (cons (car memdp) dp) 'left outs)
X      (go fail)))
X
X(de and-right (outs mem tests)
X   (prog (fp dp memdp tlist tst lind rind res)
X      (setq fp *flag-part*)
X      (setq dp *data-part*)
Xfail  (cond ((null mem) (return nil)))
X      (setq memdp (car mem))
X      (setq mem (cdr mem))
X      (setq tlist tests)
Xtloop (cond ((null tlist) (go succ)))
X      (setq tst (car tlist))
X      (setq tlist (cdr tlist))
X      (setq lind (car tlist))
X      (setq tlist (cdr tlist))
X      (setq rind (car tlist))
X      (setq tlist (cdr tlist))
X      %% the next line differs in and-left & -right
X      (setq res (tst (gelm dp rind) (gelm memdp lind)))
X      (cond (res (go tloop)) (t (go fail)))
Xsucc  %% the next line differs in and-left & -right
X      (sendto fp (cons (car dp) memdp) 'right outs)
X      (go fail)))
X
X(de teqb (new eqvar)
X   (cond
X      ((eq new eqvar) t)
X      ((not (numberp new)) nil)
X      ((not (numberp eqvar)) nil)
X      ((=alg new eqvar) t)
X      (t nil)))
X
X(put 'eq 'tb 'teqb)
X
X(de tneb (new eqvar)
X   (cond
X      ((eq new eqvar) nil)
```

```
X      ((not (numberp new)) t)
X      ((not (numberp eqvar)) t)
X      ((=alg new eqvar) nil)
X      (t t)))
X
X(put 'ne 'tb 'tneb)
X
X(de tltb (new eqvar)
X   (cond
X      ((not (numberp new)) nil)
X      ((not (numberp eqvar)) nil)
X      ((greaterp eqvar new) t)
X      (t nil)))
X
X(put 'lt 'tb 'tltb)
X
X(de tgtb (new eqvar)
X   (cond
X      ((not (numberp new)) nil)
X      ((not (numberp eqvar)) nil)
X      ((greaterp new eqvar) t)
X      (t nil)))
X
X(put 'gt 'tb 'tgtb)
X
X(de tgeb (new eqvar)
X   (cond
X      ((not (numberp new)) nil)
X      ((not (numberp eqvar)) nil)
X      ((not (greaterp eqvar new)) t)
X      (t nil)))
X
X(put 'ge 'tb 'tgeb)
X
X(de tleb (new eqvar)
X   (cond
X      ((not (numberp new)) nil)
X      ((not (numberp eqvar)) nil)
X      ((not (greaterp new eqvar)) t)
X      (t nil)))
X
X(put 'le 'tb 'tleb)
X
X(de txxb (new eqvar)
X   (cond
X      ((numberp new) (cond ((numberp eqvar) t) (t nil)))
X      (t (cond ((numberp eqvar) nil) (t t)))) )
X
X(put 'xx 'tb 'txxb)
X
X(de &p (rating name var-dope ce-var-dope rhs)
X   (prog (fp dp)
X      (cond
```

```
X     (*sendtocall* (setq fp *flag-part*) (setq dp *data-part*))
X     (t (setq fp *alpha-flag-part*) (setq dp *alpha-data-part*)))
X      (and (memq fp '(nil old)) (removecs name dp))
X      (and fp (insertcs name dp rating))))
X
X(de &old (a b c d e) nil)
X
X(de &not (outs lmem rpred tests)
X   (cond
X      ((eq *side* 'right)
X       (cond ((eq *flag-part* 'old) nil)
X         (t (not-right outs (car lmem) tests))))
X      (t (not-left outs (memory-part rpred) tests lmem))))
X
X(de not-left (outs mem tests own-mem)
X   (prog (fp dp memdp tlist tst lind rind res c)
X      (setq fp *flag-part*)
X      (setq dp *data-part*)
X      (setq c 0)
Xfail  (cond ((null mem) (go fin)))
X      (setq memdp (car mem))
X      (setq mem (cdr mem))
X      (setq tlist tests)
Xtloop (cond ((null tlist) (setq c (iadd1 c)) (go fail)))
X      (setq tst (car tlist))
X      (setq tlist (cdr tlist))
X      (setq lind (car tlist))
X      (setq tlist (cdr tlist))
X      (setq rind (car tlist))
X      (setq tlist (cdr tlist))
X      %% the next line differs in not-left & -right
X      (setq res (tst (gelm memdp rind) (gelm dp lind)))
X      (cond (res (go tloop)) (t (go fail)))
Xfin   (add-token own-mem fp dp c)
X      (cond ((izerop c) (sendto fp dp 'left outs)))))
X
X(de not-right (outs mem tests)
X   (prog (fp dp memdp tlist tst lind rind res newfp inc newc)
X      (setq fp *flag-part*)
X      (setq dp *data-part*)
X      (cond
X     ((not fp) (setq inc (!!minus 1)) (setq newfp 'new))
X     ((eq fp 'new) (setq inc 1) (setq newfp nil))
X     (t (return nil)))
Xfail  (cond ((null mem) (return nil)))
X      (setq memdp (car mem))
X      (setq newc (cadr mem))
X      (setq tlist tests)
Xtloop (cond ((null tlist) (go succ)))
X      (setq tst (car tlist))
X      (setq tlist (cdr tlist))
X      (setq lind (car tlist))
X      (setq tlist (cdr tlist))
```

```
X      (setq rind (car tlist))
X      (setq tlist (cdr tlist))
X      %% the next line differs in not-left & -right
X      (setq res (tst (gelm dp rind) (gelm memdp lind)))
X      (cond (res (go tloop)) (t (setq mem (cddr mem)) (go fail)))
Xsucc  (setq newc (iplus inc newc))
X      (rplaca (cdr mem) newc)
X      (cond
X     ((or
X         (and (eq inc (!!minus 1)) (eq newc 0))
X         (and (eq inc 1) (eq newc 1)))
X        (sendto newfp memdp 'right outs)))
X      (setq mem (cddr mem))
X      (go fail)))
X
X%%% Node memories
X
X%(de add-token (memlis flag data-part num)
X%   (prog (was-present)
X%      (cond
X%         ((eq flag 'new)
X%        (setq was-present nil)
X%            (real-add-token memlis data-part num))
X%         ((not flag)
X%            (setq was-present (remove-old memlis data-part num)))
X%         ((eq flag 'old) (setq was-present t)))
X%      (return was-present)))
X(de add-token (memlis flag data-part num)
X    (cond
X    ((eq flag 'new) (real-add-token memlis data-part num) nil)
X    ((not flag) (remove-old memlis data-part num) nil)
X    ((eq flag 'old) t)
X    (t nil)))
X
X(de real-add-token (lis data-part num)
X   (setq *current-token* (iadd1 *current-token*))
X   (cond (num (rplaca lis (cons num (car lis)))) )
X   (rplaca lis (cons data-part (car lis))))
X
X(de remove-old (lis data num)
X   (cond
X      (num (remove-old-num lis data))
X      (t (remove-old-no-num lis data))))
X
X(de remove-old-num (lis data)
X   (prog (m next last)
X      (setq m (car lis))
X      (cond
X     ((atom m) (return nil))
X     ((top-levels-eq data (car m))
X        (setq *current-token* (isub1 *current-token*))
X        (rplaca lis (cddr m))
X        (return (car m))))
```

```
X      (setq next m)
Xloop  (setq last next)
X      (setq next (cddr next))
X      (cond
X     ((atom next) (return nil))
X     ((top-levels-eq data (car next))
X        (rplacd (cdr last) (cddr next))
X        (setq *current-token* (isub1 *current-token*))
X        (return (car next)))
X     (t (go loop)))) )
X
X(de remove-old-no-num (lis data)
X   (prog (m next last)
X      (setq m (car lis))
X      (cond
X     ((atom m) (return nil))
X     ((top-levels-eq data (car m))
X        (setq *current-token* (isub1 *current-token*))
X        (rplaca lis (cdr m))
X        (return (car m))))
X      (setq next m)
Xloop  (setq last next)
X      (setq next (cdr next))
X      (cond
X     ((atom next) (return nil))
X     ((top-levels-eq data (car next))
X        (rplacd last (cdr next))
X        (setq *current-token* (isub1 *current-token*))
X        (return (car next)))
X     (t (go loop)))) )
X
X%%% Conflict Resolution
X%
X%
X% each conflict set element is a list of the following form:
X% ((p-name . data-part) (sorted wm-recency) special-case-number)
X
X(de removecs (name data)
X   (prog (cr-data inst cs)
X      (setq cr-data (cons name data))
X      (setq cs *conflict-set*)
Xl:    (cond ((null cs) (record-refract name data) (return nil)))
X      (setq inst (car cs))
X      (setq cs (cdr cs))
X      (cond ((not (top-levels-eq (car inst) cr-data)) (go l:)))
X      (setq *conflict-set* (delq inst *conflict-set*))))
X
X(de insertcs (name data rating)
X   (prog (instan)
X      (cond ((refracted name data) (return nil)))
X      (setq instan (list (cons name data) (order-tags data) rating))
X      (and (atom *conflict-set*) (setq *conflict-set* nil))
X      (return (setq *conflict-set* (cons instan *conflict-set*)))) )
```

```
X
X(de order-tags (dat)
X   (prog (tags)
X      (setq tags nil)
Xl1:   (cond ((atom dat) (go l2:)))
X      (setq tags (cons (creation-time (car dat)) tags))
X      (setq dat (cdr dat))
X      (go l1:)
Xl2:   (cond
X     ((eq *strategy* 'mea)
X        (return (cons (car tags) (dsort (cdr tags)))) )
X     (t (return (dsort tags)))) ))
X
X% destructively sort x into descending order
X(de dsort (x)
X   (prog (sorted cur next cval nval)
X      (cond ((atom (cdr x)) (return x)))
Xloop  (setq sorted t)
X      (setq cur x)
X      (setq next (cdr x))
Xchek  (setq cval (car cur))
X      (setq nval (car next))
X      (cond
X     ((greaterp nval cval)
X        (setq sorted nil)
X        (rplaca cur nval)
X        (rplaca next cval)))
X      (setq cur next)
X      (setq next (cdr cur))
X      (cond
X     ((not (null next)) (go chek))
X     (sorted (return x))
X     (t (go loop)))) )
X
X(de conflict-resolution nil
X   (prog (best len)
X      (setq len (length *conflict-set*))
X      (cond ((igreaterp len *max-cs*) (setq *max-cs* len)))
X      (setq *total-cs* (iplus *total-cs* len))
X      (cond
X     (*conflict-set*
X        (setq best (best-of *conflict-set*))
X        (setq *conflict-set* (delq best *conflict-set*))
X        (return (pname-instantiation best)))
X     (t (return nil)))) )
X
X(de best-of (set) (best-of* (car set) (cdr set)))
X
X(de best-of* (best rem)
X   (cond
X      ((not rem) best)
X      ((conflict-set-compare best (car rem))
X     (best-of* best (cdr rem)))
```

```
X      (t (best-of* (car rem) (cdr rem)))) )
X
X(de remove-from-conflict-set (name)
X   (prog (cs entry)
Xl1    (setq cs *conflict-set*)
Xl2    (cond ((atom cs) (return nil)))
X      (setq entry (car cs))
X      (setq cs (cdr cs))
X      (cond
X     ((eq name (caar entry))
X        (setq *conflict-set* (delq entry *conflict-set*))
X        (go l1))
X     (t (go l2)))) )
X
X(de pname-instantiation (conflict-elem) (car conflict-elem))
X
X(de order-part (conflict-elem) (cdr conflict-elem))
X
X(de instantiation (conflict-elem)
X   (cdr (pname-instantiation conflict-elem)))
X
X(de conflict-set-compare (x y)
X   (prog (x-order y-order xl yl xv yv)
X      (setq x-order (order-part x))
X      (setq y-order (order-part y))
X      (setq xl (car x-order))
X      (setq yl (car y-order))
Xdata  (cond
X     ((and (null xl) (null yl)) (go ps))
X     ((null yl) (return t))
X     ((null xl) (return nil)))
X      (setq xv (car xl))
X      (setq yv (car yl))
X      (cond
X     ((greaterp xv yv) (return t))
X     ((greaterp yv xv) (return nil)))
X      (setq xl (cdr xl))
X      (setq yl (cdr yl))
X      (go data)
Xps    (setq xl (cdr x-order))
X      (setq yl (cdr y-order))
Xpsl   (cond ((null xl) (return t)))
X      (setq xv (car xl))
X      (setq yv (car yl))
X      (cond
X     ((greaterp xv yv) (return t))
X     ((greaterp yv xv) (return nil)))
X      (setq xl (cdr xl))
X      (setq yl (cdr yl))
X      (go psl)))
X
X(de conflict-set nil
X   (prog (cnts cs p z best)
```

```
X      (setq cnts nil)
X      (setq cs *conflict-set*)
Xl1:   (cond ((atom cs) (go l2:)))
X      (setq p (caaar cs))
X      (setq cs (cdr cs))
X      (setq z (atsoc p cnts))
X      (cond
X     ((null z) (setq cnts (cons (cons p 1) cnts)))
X     (t (rplacd z (iadd1 (cdr z)))) )
X      (go l1:)
Xl2:   (cond
X     ((atom cnts)
X        (setq best (best-of *conflict-set*))
X        (terpri)
X        (return (list (caar best) 'dominates))))
X      (terpri)
X      (princ (caar cnts))
X      (cond
X     ((greaterp (cdar cnts) 1)
X        (princ "        (")
X        (princ (cdar cnts))
X        (princ " occurrences)")))
X      (setq cnts (cdr cnts))
X      (go l2:)))
X
X%%% WM maintaining functions
X%
X% The order of operations in the following two functions is critical.
X% add-to-wm order: (1) change wm (2) record change (3) match
X% remove-from-wm order: (1) record change (2) match (3) change wm
X% (back will not restore state properly unless wm changes are recorded
X% before the cs changes that they cause)  (match will give errors if
X% the thing matched is not in wm at the time)
X
X(de add-to-wm (wme override)
X   (prog (fa z part timetag port)
X      (setq *critical* t)
X      (setq *current-wm* (iadd1 *current-wm*))
X      (and
X     (greaterp *current-wm* *max-wm*)
X     (setq *max-wm* *current-wm*))
X      (setq *action-count* (iadd1 *action-count*))
X      (setq fa (wm-hash wme))
X      (or
X     (memq fa *wmpart-list*)
X     (setq *wmpart-list* (cons fa *wmpart-list*)))
X      (setq part (get fa 'wmpart*))
X      (cond
X     (override (setq timetag override))
X     (t (setq timetag *action-count*)))
X      (setq z (cons wme timetag))
X      (putprop fa (cons z part) 'wmpart*)
X      (record-change '=>wm *action-count* wme)
```

```
X      (match 'new wme)
X      (setq *critical* nil)
X      (cond
X     ((and *in-rhs* *wtrace*)
X        (setq port (trace-file))
X        (terpri port)
X        (!!princ "=>wm: " port)
X        (ppelm wme port)))) )
X
X% remove-from-wm uses eq, not equal to determine if wme is present
X(de remove-from-wm (wme)
X   (prog (fa z part timetag port)
X      (setq fa (wm-hash wme))
X      (setq part (get fa 'wmpart*))
X      (setq z (atsoc wme part))
X      (cond ((null z) (return nil)))
X      (setq timetag (cdr z))
X      (cond
X     ((and *wtrace* *in-rhs*)
X        (setq port (trace-file))
X        (terpri port)
X        (!!princ "<=wm: " port)
X        (ppelm wme port)))
X      (setq *action-count* (iadd1 *action-count*))
X      (setq *critical* t)
X      (setq *current-wm* (sub1 *current-wm*))
X      (record-change '<=wm timetag wme)
X      (match nil wme)
X      (putprop fa (delq z part) 'wmpart*)
X      (setq *critical* nil)))
X
X% mapwm maps down the elements of wm, applying fn to each element
X% each element is of form (datum . creation-time)
X(de mapwm (fn)
X   (prog (wmpl part)
X      (setq wmpl *wmpart-list*)
Xlab1  (cond ((atom wmpl) (return nil)))
X      (setq part (get (car wmpl) 'wmpart*))
X      (setq wmpl (cdr wmpl))
X      (!!mapc fn part)
X      (go lab1)))
X
X(df wm a
X   (!!mapc (function (lambda (z) (terpri) (ppelm z nil))) (get-wm a))
X   nil)
X
X(de get-wm (z)
X   (setq *wm-filter* z)
X   (setq *wm* nil)
X   (mapwm (function get-wm2))
X   (prog1 *wm* (setq *wm* nil)))
X
X(de get-wm2 (elem)
```

```
X   (cond
X      ((or (null *wm-filter*) (member (cdr elem) *wm-filter*))
X     (setq *wm* (cons (car elem) *wm*)))) )
X
X(de wm-hash (x)
X   (cond
X      ((not x) '<default>)
X      ((not (car x)) (wm-hash (cdr x)))
X      ((idp (car x)) (car x))
X      (t (wm-hash (cdr x)))) )
X
X(de creation-time (wme)
X   (cdr (atsoc wme (get (wm-hash wme) 'wmpart*))))
X
X(de refresh nil
X   (prog nil
X      (setq *old-wm* nil)
X      (mapwm (function refresh-collect))
X      (!!mapc (function refresh-del) *old-wm*)
X      (!!mapc (function refresh-add) *old-wm*)
X      (setq *old-wm* nil)))
X
X(de refresh-collect (x) (setq *old-wm* (cons x *old-wm*)))
X
X(de refresh-del (x) (remove-from-wm (car x)))
X
X(de refresh-add (x) (add-to-wm (car x) (cdr x)))
X
X(de trace-file ()
X   (prog (port)
X      (setq port nil)
X      (cond
X     (*trace-file*
X        (setq port ($ofile *trace-file*))
X        (cond
X           ((null port)
X           (!%warn "trace: file has been closed" *trace-file*)
X           (setq port nil)))) )
X      (return port)))
X
X%%% Basic functions for RHS evaluation
X
X(de eval-rhs (pname data)
X   (prog (node port)
X      (cond
X     (*ptrace*
X        (setq port (trace-file))
X        (terpri port)
X        (!!princ *cycle-count* port)
X        (!!princ ". " port)
X        (!!princ pname port)
X        (time-tag-print data port)))
X      (setq *data-matched* data)
```

```
X      (setq *p-name* pname)
X      (setq *last* nil)
X      (setq node (get pname 'topnode))
X      (init-var-mem (var-part node))
X      (init-ce-var-mem (ce-var-part node))
X      (begin-record pname data)
X      (setq *in-rhs* t)
X      (eval (rhs-part node))
X      (setq *in-rhs* nil)
X      (end-record)))
X
X(de time-tag-print (data port)
X   (cond
X      ((not (null data))
X     (time-tag-print (cdr data) port)
X     (!!princ " " port)
X     (!!princ (creation-time (car data)) port))))
X
X(de init-var-mem (vlist)
X   (prog (v ind r)
X      (setq *variable-memory* nil)
Xtop   (cond ((atom vlist) (return nil)))
X      (setq v (car vlist))
X      (setq ind (cadr vlist))
X      (setq vlist (cddr vlist))
X      (setq r (gelm *data-matched* ind))
X      (setq *variable-memory* (cons (cons v r) *variable-memory*))
X      (go top)))
X
X(de init-ce-var-mem (vlist)
X   (prog (v ind r)
X      (setq *ce-variable-memory* nil)
Xtop   (cond ((atom vlist) (return nil)))
X      (setq v (car vlist))
X      (setq ind (cadr vlist))
X      (setq vlist (cddr vlist))
X      (setq r (ce-gelm *data-matched* ind))
X      (setq *ce-variable-memory*
X     (cons (cons v r) *ce-variable-memory*))
X      (go top)))
X
X(de make-ce-var-bind (var elem)
X   (setq *ce-variable-memory*
X      (cons (cons var elem) *ce-variable-memory*)))
X
X(de make-var-bind (var elem)
X   (setq *variable-memory* (cons (cons var elem) *variable-memory*)))
X
X(de $varbind (x)
X   (prog (r)
X      (cond ((not *in-rhs*) (return x)))
X      (setq r (atsoc x *variable-memory*))
X      (cond (r (return (cdr r))) (t (return x))) )
```

```
X
X(de get-ce-var-bind (x)
X   (prog (r)
X      (cond ((numberp x) (return (get-num-ce x))))
X      (setq r (atsoc x *ce-variable-memory*))
X      (cond (r (return (cdr r))) (t (return nil))) )
X
X(de get-num-ce (x)
X   (prog (r l d)
X      (setq r *data-matched*)
X      (setq l (length r))
X      (setq d (difference l x))
X      (cond ((greaterp 0 d) (return nil)))
Xla    (cond
X     ((null r) (return nil))
X     ((greaterp 1 d) (return (car r))))
X      (setq d (sub1 d))
X      (setq r (cdr r))
X      (go la)))
X
X(de build-collect (z)
X   (prog (r)
Xla    (cond ((atom z) (return nil)))
X      (setq r (car z))
X      (setq z (cdr z))
X      (cond
X     ((pairp r) ($value '!() (build-collect r) ($value '!) ))
X     ((eq r '!!) ($change (car z)) (setq z (cdr z)))
X     (t ($value r)))
X      (go la)))
X
X(de unflat (x) (setq *rest* x) (unflat*))
X
X(de unflat* nil
X   (prog (c)
X      (cond ((atom *rest*) (return nil)))
X      (setq c (car *rest*))
X      (setq *rest* (cdr *rest*))
X      (cond
X     ((eq c '!() (return (cons (unflat*) (unflat*))))
X     ((eq c '!)) (return nil))
X     (t (return (cons c (unflat*)))) )))
X
X(de $change (x)
X   (prog nil
X      (cond
X     ((pairp x) (eval-function x))
X     (t ($value ($varbind x)))) ))
X
X(de eval-args (z)
X   (prog (r)
X      (rhs-tab 1)
Xla    (cond ((atom z) (return nil)))
```

```
X      (setq r (car z))
X      (setq z (cdr z))
X      (cond
X     ((eq r '!^)
X       (rhs-tab (car z))
X       (setq r (cadr z))
X       (setq z (cddr z))))
X      (cond
X     ((eq r '!/) ($value (car z)) (setq z (cdr z)))
X     (t ($change r)))
X      (go la)))
X
X(de eval-function (form)
X   (cond
X      ((not *in-rhs*)
X     (!%warn "functions cannot be used at top level" (car form)))
X      (t (eval form))))
X
X
X%%% Functions to manipulate the result array
X
X(de $reset nil (setq *max-index* 0) (setq *next-index* 1))
X
X% rhs-tab implements the tab ('^') function in the rhs.  it has
X% four responsibilities:
X%       - to move the array pointers
X%       - to watch for tabbing off the left end of the array
X%         (ie, to watch for pointers less than 1)
X%       - to watch for tabbing off the right end of the array
X%       - to write nil in all the slots that are skipped
X% the last is necessary if the result array is not to be cleared
X% after each use% if rhs-tab did not do this, $reset
X% would be much slower.
X
X(de rhs-tab (z) ($tab ($varbind z)))
X
X(de $tab (z)
X   (prog (edge next)
X      (setq next ($litbind z))
X      (and (floatp next) (setq next (fix next)))
X      (cond
X     ((or
X         (not (numberp next))
X         (greaterp next *size-result-array*)
X         (greaterp 1 next))
X       (!%warn "illegal index after ^" next)
X       (return *next-index*)))
X      (setq edge (isub1 next))
X      (cond ((greaterp *max-index* edge) (go ok)))
Xclear (cond ((eq *max-index* edge) (go ok)))
X      (putv *result-array* edge nil)
X      (setq edge (isub1 edge))
X      (go clear)
```

```
Xok    (setq *next-index* next)
X      (return next)))
X
X(de $value (v)
X   (cond
X      ((greaterp *next-index* *size-result-array*)
X     (!%warn "index too large" *next-index*))
X      (t (and
X        (greaterp *next-index* *max-index*)
X        (setq *max-index* *next-index*))
X     (putv *result-array* *next-index* v)
X     (setq *next-index* (iadd1 *next-index*)))) )
X
X(de use-result-array nil
X   (prog (k r)
X      (setq k *max-index*)
X      (setq r nil)
Xtop   (cond ((eq k 0) (return r)))
X      (setq r (cons (getv *result-array* k) r))
X      (setq k (isub1 k))
X      (go top)))
X
X(de $assert nil
X   (setq *last* (use-result-array))
X   (add-to-wm *last* nil))
X
X(de $parametercount nil *max-index*)
X
X(de $parameter (k)
X   (cond
X      ((or
X      (not (numberp k))
X      (igreaterp k *size-result-array*)
X      (ilessp k 1))
X     (!%warn "illegal parameter number " k)
X     nil)
X      ((igreaterp k *max-index*) nil)
X      (t (getv *result-array* k))))
X
X%%% RHS actions
X
X(df make z
X   (prog nil
X      ($reset)
X      (eval-args z)
X      ($assert)))
X
X(df modify z
X   (prog (old)
X      (cond
X     ((not *in-rhs*)
X        (!%warn "cannot be called at top level" 'modify)
X        (return nil)))
```

```
X      (setq old (get-ce-var-bind (car z)))
X      (cond
X     ((null old)
X        (!%warn
X           "modify: first argument must be an element variable"
X           (car z))
X        (return nil)))
X      (remove-from-wm old)
X      (setq z (cdr z))
X      ($reset)
Xcopy   (cond ((atom old) (go fin)))
X      ($change (car old))
X      (setq old (cdr old))
X      (go copy)
Xfin    (eval-args z)
X      ($assert)))
X
X(df bind z
X   (prog (val)
X      (cond
X     ((not *in-rhs*)
X        (!%warn "cannot be called at top level" 'bind)
X        (return nil)))
X      (cond
X     ((ilessp (length z) 1)
X        (!%warn "bind: wrong number of arguments to" z)
X        (return nil))
X     ((not (idp (car z)))
X        (!%warn "bind: illegal argument" (car z))
X        (return nil))
X     ((eq (length z) 1) (setq val (gensym)))
X     (t ($reset) (eval-args (cdr z)) (setq val ($parameter 1))))
X      (make-var-bind (car z) val)))
X
X(df cbind z
X   (cond
X      ((not *in-rhs*)
X     (!%warn "cannot be called at top level" 'cbind))
X      ((not (eq (length z) 1))
X     (!%warn "cbind: wrong number of arguments" z))
X      ((not (idp (car z)))
X     (!%warn "cbind: illegal argument" (car z)))
X      ((null *last*) (!%warn "cbind: nothing added yet" (car z)))
X      (t (make-ce-var-bind (car z) *last*))))
X
X(df remove z
X   (prog (old)
X      (cond ((not *in-rhs*) (return (top-level-remove z))))
Xtop    (cond ((atom z) (return nil)))
X      (setq old (get-ce-var-bind (car z)))
X      (cond
X     ((null old)
X        (!%warn
```

```
X           "remove: argument not an element variable"
X           (car z))
X         (return nil)))
X       (remove-from-wm old)
X       (setq z (cdr z))
X       (go top)))
X
X(df call z
X   (prog (f)
X       (setq f (car z))
X       ($reset)
X       (eval-args (cdr z))
X       (f)))
X
X(df write z
X   (prog (port max k x needspace)
X       (cond
X     ((not *in-rhs*)
X        (!%warn "cannot be called at top level" 'write)
X        (return nil)))
X       ($reset)
X       (eval-args z)
X       (setq k 1)
X       (setq max ($parametercount))
X       (cond
X     ((ilessp max 1)
X        (!%warn "write: nothing to print" z)
X        (return nil)))
X       (setq port (default-write-file))
X       (setq x ($parameter 1))
X       (cond
X     ((and (idp x) ($ofile x))
X        (setq port ($ofile x))
X        (setq k 2)))
X       (setq needspace t)
Xla     (cond ((greaterp k max) (return nil)))
X       (setq x ($parameter k))
X       (cond
X     ((eq x "=== C R L F ===")
X        (setq needspace nil)
X        (terpri port))
X     ((eq x "=== R J U S T ===")
X        (setq k (iplus 2 k))
X        (do-rjust ($parameter (isub1 k)) ($parameter k) port))
X     ((eq x "=== T A B T O ===")
X        (setq needspace nil)
X        (setq k (iadd1 k))
X        (do-tabto ($parameter k) port))
X     (t (and needspace (!!princ " " port))
X        (setq needspace t)
X        (!!princ x port)))
X       (setq k (iadd1 k))
X       (go la)))
```

```
X
X(de default-write-file ()
X   (prog (port)
X      (setq port nil)
X      (cond
X     (*write-file*
X        (setq port ($ofile *write-file*))
X        (cond
X           ((null port)
X           (!%warn "write: file has been closed" *write-file*)
X           (setq port nil)))) )
X      (return port)))
X
X(de do-rjust (width value port k)
X   (prog (size)
X      (cond
X     ((eq value "=== T A B T O ===")
X        (!%warn "rjust cannot precede this function" 'tabto)
X        (return nil))
X     ((eq value "=== C R L F ===")
X        (!%warn "rjust cannot precede this function" 'crlf)
X        (return nil))
X     ((eq value "=== R J U S T ===")
X        (!%warn "rjust cannot precede this function" 'rjust)
X        (return nil)))
X      (setq size (flatc value (iadd1 width)))
X      (cond
X     ((greaterp size width)
X        (!!princ " " port)
X        (!!princ value port)
X        (return nil)))
X      (setq k (difference width size))
X      (while (greaterp k 0)
X     (progn (setq k (isub1 k))
X         (!!princ " " port)))
X      (!!princ value port)))
X
X(de do-tabto (col port)
X   (prog (pos k)
X      (setq pos (iadd1 (posn port)))
X      (cond ((greaterp pos col) (terpri port) (setq pos 1)))
X      (setq k (difference col pos))
X      (while (greaterp k 0)
X     (progn (setq k (isub1 k))
X        (!!princ " " port)))
X      (return nil)))
X
X(de halt nil
X   (cond
X      ((not *in-rhs*) (!%warn "cannot be called at top level" 'halt))
X      (t (setq *halt-flag* t))))
X
X(de build z
```

```
X   (prog (r)
X      (cond
X     ((not *in-rhs*)
X        (!%warn "cannot be called at top level" 'build)
X        (return nil)))
X      ($reset)
X      (build-collect z)
X      (setq r (unflat (use-result-array)))
X      (and *build-trace* (*build-trace* r))
X      (compile-production (car r) (cdr r))))
X
X(df openfile z
X   (prog (file mode id)
X      ($reset)
X      (eval-args z)
X      (cond
X     ((not (eq ($parametercount) 3))
X        (!%warn "openfile: wrong number of arguments" z)
X        (return nil)))
X      (setq id ($parameter 1))
X      (setq file ($parameter 2))
X      (setq mode ($parameter 3))
X      (cond
X     ((not (idp id))
X        (!%warn "openfile: file id must be a symbolic atom" id)
X        (return nil))
X     ((null id)
X        (!%warn
X           "openfile: 'nil' is reserved for the terminal"
X           nil)
X        (return nil))
X     ((or ($ifile id) ($ofile id))
X        (!%warn "openfile: name already in use" id)
X        (return nil)))
X      (cond
X     ((eq mode 'in) (putprop id (open file 'input) 'inputfile))
X     ((eq mode 'out) (putprop id (open file 'output) 'outputfile))
X     (t (!%warn "openfile: illegal mode" mode) (return nil)))
X      (return nil)))
X
X(de $ifile (x) (get x 'inputfile))
X
X(de $ofile (x) (get x 'outputfile))
X
X(df closefile z
X   ($reset)
X   (eval-args z)
X   (!!mapc (function closefile2) (use-result-array)))
X
X(de closefile2 (file)
X   (prog (port)
X      (cond
X     ((not (idp file))
```

```
X       (!%warn "closefile: illegal file identifier" file))
X     ((setq port ($ifile file))
X       (close port)
X       (remprop file 'inputfile))
X     ((setq port ($ofile file))
X       (close port)
X       (remprop file 'outputfile)))
X      (return nil)))
X
X(df default z
X   (prog (file use)
X      ($reset)
X      (eval-args z)
X      (cond
X     ((not (eq ($parametercount) 2))
X       (!%warn "default: wrong number of arguments" z)
X       (return nil)))
X      (setq file ($parameter 1))
X      (setq use ($parameter 2))
X      (cond
X     ((not (idp file))
X       (!%warn "default: illegal file identifier" file)
X       (return nil))
X     ((not (memq use '(write accept trace)))
X       (!%warn "default: illegal use for a file" use)
X       (return nil))
X     ((and
X         (memq use '(write trace))
X         (not (null file))
X         (not ($ofile file)))
X       (!%warn
X          "default: file has not been opened for output"
X          file)
X       (return nil))
X     ((and
X         (eq use 'accept)
X         (not (null file))
X         (not ($ifile file)))
X       (!%warn
X          "default: file has not been opened for input"
SHAR_EOF
echo "End of part 2, continue with part 3"
echo "3" > s2_seq_.tmp
exit 0


From -8764465151521866505
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,6b975f32c2711a56,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-06-01 13:53:11 PST
Path: utzoo!utgpu!water!watmath!clyde!att!osu-cis!tut.cis.ohio-state.edu!bloom-
beacon!mit-eddie!ll-xn!ames!pasteur!agate!saturn!ssyx!koreth
```

From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i051: ops5 -- OPS5 System in Cambridge Lisp part03/03
Keywords: shar, lisp
Message-ID: <3563@saturn.ucsc.edu>
Date: 1 Jun 88 20:53:11 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1049
Approved: koreth@ssyx.ucsc.edu

Submitted-by: cfc@wjh12.harvard.edu (Christopher F. Chabris)
Posting-number: Volume 1, Issue 51
Archive-name: ops5/part03

```
#!/bin/sh
# this is part 3 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file ops5 continued
#
CurArch=3
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> ops5
X           file)
X       (return nil))
X     ((eq use 'write) (setq *write-file* file))
X     ((eq use 'accept) (setq *accept-file* file))
X     ((eq use 'trace) (setq *trace-file* file)))
X      (return nil)))
X
X
X%%% RHS Functions
X
X(df accept z
X   (prog (port arg)
X      (cond
X     ((igreaterp (length z) 1)
X        (!%warn "accept: wrong number of arguments" z)
X        (return nil)))
X      (setq port nil)
X      (cond
X     (*accept-file*
X        (setq port ($ifile *accept-file*))
X        (cond
X           ((null port)
X           (!%warn
```

```
X              "accept: file has been closed"
X              *accept-file*)
X           (return nil)))) )
X      (cond
X     ((eq (length z) 1)
X        (setq arg ($varbind (car z)))
X        (cond
X           ((not (idp arg))
X           (!%warn "accept: illegal file name" arg)
X           (return nil)))
X        (setq port ($ifile arg))
X        (cond
X           ((null port)
X           (!%warn "accept: file not open for input" arg)
X           (return nil)))) )
X      (cond
X     ((eq (!!tyipeek port) (!!minus 1))
X        ($value 'end-of-file)
X        (return nil)))
X      (flat-value (!!read port))))
X
X(de flat-value (x)
X   (cond ((atom x) ($value x)) (t (!!mapc (function flat-value) x))))
X
X(de span-chars (x prt)
X (prog (ch)
X   (setq ch (!!tyipeek prt))
X   (while (member ch x)
X      (progn (!!readc prt) (setq ch (!!tyipeek prt))))))
X
X(df acceptline z
X   (prog (c def arg port)
X      (setq port nil)
X      (setq def z)
X      (cond
X     (*accept-file*
X        (setq port ($ifile *accept-file*))
X        (cond
X           ((null port)
X           (!%warn
X              "acceptline: file has been closed"
X              *accept-file*)
X           (return nil)))) )
X      (cond
X     ((pairp def)       %% replaces the awful (greaterp (length def) 0)
X        (setq arg ($varbind (car def)))
X        (cond
X           ((and (idp arg) ($ifile arg))
X           (setq port ($ifile arg))
X           (setq def (cdr def)))) ))
X      (span-chars '(9 41) port)
X      (setq c (tyi port))
X      (cond
```

```
X     ((memq (!!tyipeek port) '(-1 10))
X        (!!mapc (function $change) def)
X        (return nil)))
Xl:    (flat-value (!!read port))
X      (span-chars '(9 41) port)
X      (cond
X     ((not (memq (!!tyipeek port) '(difference1 10)))
X        (go l:)))) )
X
X(df substr l
X   (prog (k elm start end)
X      (cond
X     ((not (eq (length l) 3))
X        (!%warn "substr: wrong number of arguments" l)
X        (return nil)))
X      (setq elm (get-ce-var-bind (car l)))
X      (cond
X     ((null elm)
X        (!%warn "first argument to substr must be a ce var" l)
X        (return nil)))
X      (setq start ($varbind (cadr l)))
X      (setq start ($litbind start))
X      (cond
X     ((not (numberp start))
X        (!%warn "second argument to substr must be a number" l)
X        (return nil)))
X      %%   if a variable is bound to INF, the following
X      %%   will get the binding and treat it as INF is
X      %%   always treated.  That may not be good.
X      (setq end ($varbind (caddr l)))
X      (cond ((eq end 'inf) (setq end (length elm))))
X      (setq end ($litbind end))
X      (cond
X     ((not (numberp end))
X        (!%warn "third argument to substr must be a number" l)
X        (return nil)))
X      %%   this loop does not check for the end of elm
X      %%   instead it relies on cdr of nil being nil
X      %%   this may not work in all versions of lisp
X      (setq k 1)
Xla    (cond
X     ((igreaterp k end) (return nil))
X     ((not (ilessp k start)) ($value (car elm))))
X      (setq elm (cdr elm))
X      (setq k (iadd1 k))
X      (go la)))
X
X(df compute z ($value (ari z)))
X
X% arith is the obsolete form of compute
X(df arith z ($value (ari z)))
X
X(de ari (x)
```

```
X   (cond
X      ((atom x) (!%warn "bad syntax in arithmetic expression " x) 0)
X      ((atom (cdr x)) (ari-unit (car x)))
X      ((eq (cadr x) '+) (plus (ari-unit (car x)) (ari (cddr x))))
X      ((eq (cadr x) '-)
X     (difference (ari-unit (car x)) (ari (cddr x))))
X      ((eq (cadr x) '*) (times (ari-unit (car x)) (ari (cddr x))))
X      ((eq (cadr x) '!/)
X     (quotient (ari-unit (car x)) (ari (cddr x))))
X      ((eq (cadr x) '!!)
X     (mod (fix (ari-unit (car x))) (fix (ari (cddr x)))) )
X      (t (!%warn "bad syntax in arithmetic expression " x) 0)))
X
X(de ari-unit (a)
X   (prog (r)
X      (cond ((pairp a) (setq r (ari a))) (t (setq r ($varbind a))))
X      (cond
X     ((not (numberp r))
X        (!%warn "bad value in arithmetic expression" a)
X        (return 0))
X     (t (return r)))) )
X
X(de genatom nil ($value (gensym)))
X
X(df litval z
X   (prog (r)
X      (cond
X     ((not (eq (length z) 1))
X        (!%warn "litval: wrong number of arguments" z)
X        ($value 0)
X        (return nil))
X     ((numberp (car z)) ($value (car z)) (return nil)))
X      (setq r ($litbind ($varbind (car z))))
X      (cond ((numberp r) ($value r) (return nil)))
X      (!%warn "litval: argument has no literal binding" (car z))
X      ($value 0)))
X
X(df rjust z
X   (prog (val)
X      (cond
X     ((not (eq (length z) 1))
X        (!%warn "rjust: wrong number of arguments" z)
X        (return nil)))
X      (setq val ($varbind (car z)))
X      (cond
X     ((or (not (numberp val)) (ilessp val 1) (igreaterp val 127))
X        (!%warn "rjust: illegal value for field width" val)
X        (return nil)))
X      ($value "=== R J U S T ===")
X      ($value val)))
X
X(df crlf z
X   (cond
```

```
X      (z (!%warn "crlf: does not take arguments" z))
X      (t ($value "=== C R L F ===")))
X
X(df tabto z
X   (prog (val)
X      (cond
X     ((not (eq (length z) 1))
X        (!%warn "tabto: wrong number of arguments" z)
X        (return nil)))
X      (setq val ($varbind (car z)))
X      (cond
X     ((or (not (numberp val)) (ilessp val 1) (igreaterp val 127))
X        (!%warn "tabto: illegal column number" z)
X        (return nil)))
X      ($value "=== T A B T O ===")
X      ($value val)))
X
X%%% Printing WM
X
X(df ppwm avlist
X   (prog (next a)
X      (setq *filters* nil)
X      (setq next 1)
Xl:    (cond ((atom avlist) (go print)))
X      (setq a (car avlist))
X      (setq avlist (cdr avlist))
X      (cond
X     ((eq a '!^)
X        (setq next (car avlist))
X        (setq avlist (cdr avlist))
X        (setq next ($litbind next))
X        (and (floatp next) (setq next (fix next)))
X        (cond
X           ((or
X            (not (numberp next))
X            (igreaterp next *size-result-array*)
X            (igreaterp 1 next))
X           (!%warn "illegal index after ^" next)
X           (return nil))))
X     ((variablep a)
X        (!%warn "ppwm does not take variables" a)
X        (return nil))
X     (t (setq *filters* (cons next (cons a *filters*)))
X        (setq next (iadd1 next))))
X      (go l:)
Xprint (mapwm (function ppwm2))
X      (terpri)
X      (return nil)))
X
X(de ppwm2 (elm-tag)
X   (cond ((filter (car elm-tag)) (terpri) (ppelm (car elm-tag) nil))))
X
X(de filter (elm)
```

```
X   (prog (fl indx val)
X      (setq fl *filters*)
Xtop   (cond ((atom fl) (return t)))
X      (setq indx (car fl))
X      (setq val (cadr fl))
X      (setq fl (cddr fl))
X      (cond ((ident (nth (isub1 indx) elm) val) (go top)))
X      (return nil)))
X
X(de ident (x y)
X   (cond
X      ((eq x y) t)
X      ((not (numberp x)) nil)
X      ((not (numberp y)) nil)
X      ((=alg x y) t)
X      (t nil)))
X
X% the new ppelm is designed especially to handle literalize format
X% however, it will do as well as the old ppelm on other formats
X(de ppelm (elm port)
X   (prog (ppdat sep val att mode lastpos curpos vlist)
X      (!!princ (creation-time elm) port)
X      (!!princ ":  " port)
X      (setq mode 'vector)
X      (setq ppdat (get (car elm) 'ppdat))
X      (and ppdat (setq mode 'a-v))
X      (setq sep "(")
X      (setq lastpos 0)
X      (setq curpos 1) (setq vlist elm)
X      (while (not (atom vlist))
X       (progn
X     (setq val (car vlist))
X     (setq att (assoc curpos ppdat))
X     (cond (att (setq att (cdr att))) (t (setq att curpos)))
X     (and
X        (idp att)
X        (is-vector-attribute att)
X        (setq mode 'vector))
X     (cond
X        ((or (not (null val)) (eq mode 'vector))
X           (!!princ sep port)
X           (ppval val att lastpos port)
X           (setq sep "    ")
X           (setq lastpos curpos)))
X     (setq curpos (iadd1 curpos))
X     (setq vlist (cdr vlist))))
X      (!!princ ")" port)))
X
X(de ppval (val att lastpos port)
X   (cond
X      ((not (eq att (iadd1 lastpos)))
X     (!!princ '!^ port)
X     (!!princ att port)
```

```
X     (!!princ " " port)))
X   (!!princ val port))
X
X%%% printing production memory
X
X(df pm z (!!mapc (function pprule) z) (terpri) nil)
X
X(de pprule (name)
X   (prog (matrix next lab)
X      (cond ((not (idp name)) (return nil)))
X      (setq matrix (get name 'production))
X      (cond ((null matrix) (return nil)))
X      (terpri)
X      (princ "(p ")
X      (princ name)
Xtop   (cond ((atom matrix) (go fin)))
X      (setq next (car matrix))
X      (setq matrix (cdr matrix))
X      (setq lab nil)
X      (terpri)
X      (cond
X     ((eq next '-)
X        (princ "  - ")
X        (setq next (car matrix))
X        (setq matrix (cdr matrix)))
X     ((eq next '-->) (princ "  "))
X     ((and (eq next '!{) (atom (car matrix)))
X        (princ "   {")
X        (setq lab (car matrix))
X        (setq next (cadr matrix))
X        (setq matrix (cdddr matrix)))
X     ((eq next '!{)
X        (princ "   {")
X        (setq lab (cadr matrix))
X        (setq next (car matrix))
X        (setq matrix (cdddr matrix)))
X     (t (princ "    ")))
X      (ppline next)
X      (cond (lab (princ " ") (princ lab) (princ '!})))
X      (go top)
Xfin   (princ ")")))
X
X(de ppline (line)
X   (prog ()
X      (cond
X     ((atom line) (princ line))
X     (t (princ "(") (setq *ppline* line) (ppline2) (princ ")")))
X      (return nil)))
X
X(de ppline2 ()
X   (prog (needspace)
X      (setq needspace nil)
Xtop   (cond ((atom *ppline*) (return nil)))
```

```
X      (and needspace (princ " "))
X      (cond ((eq (car *ppline*) '!^) (ppattval)) (t (pponlyval)))
X      (setq needspace t)
X      (go top)))
X
X(de ppattval ()
X   (prog (att val)
X      (setq att (cadr *ppline*))
X      (setq *ppline* (cddr *ppline*))
X      (setq val (getval))
X      (cond
X     ((greaterp (iplus (posn) (flatc att) (flatc val)) 76)
X        (terpri)
X        (princ "        ")))
X      (princ '!^)
X      (princ att)
X      (!!mapc (function (lambda (z) (princ " ") (princ z))) val)))
X
X(de pponlyval ()
X   (prog (val needspace)
X      (setq val (getval))
X      (setq needspace nil)
X      (cond
X     ((greaterp (iplus (posn) (flatc val)) 76)
X        (setq needspace nil)
X        (terpri)
X        (princ "        ")))
Xtop   (cond ((atom val) (return nil)))
X      (and needspace (princ " "))
X      (setq needspace t)
X      (princ (car val))
X      (setq val (cdr val))
X      (go top)))
X
X(de getval ()
X   (prog (res v1)
X      (setq v1 (car *ppline*))
X      (setq *ppline* (cdr *ppline*))
X      (cond
X     ((memq v1 '(= <> < <= => > <=>))
X        (setq res (cons v1 (getval))))
X     ((eq v1 '!{) (setq res (cons v1 (getupto '!})))
X     ((eq v1 '<<) (setq res (cons v1 (getupto '>>))))
X     ((eq v1 '!/)
X        (setq res (list v1 (car *ppline*)))
X        (setq *ppline* (cdr *ppline*)))
X     (t (setq res (list v1))))
X      (return res)))
X
X(de getupto (end)
X   (prog (v)
X      (cond ((atom *ppline*) (return nil)))
X      (setq v (car *ppline*))
```

```
X      (setq *ppline* (cdr *ppline*))
X      (cond
X     ((eq v end) (return (list v)))
X     (t (return (cons v (getupto end)))) )))
X
X
X%%% backing up
X
X(de record-index-plus (k)
X   (setq *record-index* (iplus k *record-index*))
X   (cond
X      ((lessp *record-index* 0)
X     (setq *record-index* *max-record-index*))
X      ((greaterp *record-index* *max-record-index*)
X     (setq *record-index* 0))))
X
X% the following routine initializes the record.  putting nil in the
X% first slot indicates that that the record does not go back further
X% than that.  (when the system backs up, it writes nil over the used
X% records so that it will recognize which records it has used.  thus
X% the system is set up anyway never to back over a nil.)
X(de initialize-record nil
X   (setq *record-index* 0)
X   (setq *recording* nil)
X   (setq *max-record-index* 31)
X   (putv *record-array* 0 nil))
X
X% *max-record-index* holds the maximum legal index for record-array
X% so it and the following must be changed at the same time
X(de begin-record (p data)
X   (setq *recording* t)
X   (setq *record* (list '=>refract p data)))
X
X(de end-record nil
X   (cond
X      (*recording*
X     (setq *record*
X        (cons *cycle-count* (cons *p-name* *record*)))
X     (record-index-plus 1)
X     (putv *record-array* *record-index* *record*)
X     (setq *record* nil)
X     (setq *recording* nil))))
X
X(de record-change (direct time elm)
X   (cond
X      (*recording*
X     (setq *record*
X        (cons direct (cons time (cons elm *record*)))) )))
X
X% to maintain refraction information, need keep only one piece of information:
X% need to record all unsuccessful attempts to delete things from the conflict
X% set.  unsuccessful deletes are caused by attempting to delete refracted
X% instantiations.  when backing up, have to avoid putting things back into the
```

```
X% conflict set if they were not deleted when running forward
X(de record-refract (rule data)
X   (and
X      *recording*
X      (setq *record*
X     (cons '<=refract (cons rule (cons data *record*)))) ))
X
X(de refracted (rule data)
X   (prog (z)
X      (cond ((null *refracts*) (return nil)))
X      (setq z (cons rule data))
X      (return (member z *refracts*))))
X
X(de back (k)
X   (prog (r)
Xl:    (cond ((lessp k 1) (return nil)))
X      (setq r (getv *record-array* *record-index*))
X      (cond ((null r) (return "nothing more stored")))
X      (putv *record-array* *record-index* nil)
X      (record-index-plus (!!minus 1))
X      (undo-record r)
X      (setq k (isub1 k))
X      (go l:)))
X
X(de undo-record (r)
X   (prog (save act a b rate)
X      %% *recording* must be off during back up
X      (setq save *recording*)
X      (setq *refracts* nil)
X      (setq *recording* nil)
X      (and *ptrace* (back-print (list 'undo: (car r) (cadr r))))
X      (setq r (cddr r))
Xtop   (cond ((atom r) (go fin)))
X      (setq act (car r))
X      (setq a (cadr r))
X      (setq b (caddr r))
X      (setq r (cdddr r))
X      (and *wtrace* (back-print (list 'undo: act a)))
X      (cond
X     ((eq act '<=wm) (add-to-wm b a))
X     ((eq act '=>wm) (remove-from-wm b))
X     ((eq act '<=refract)
X        (setq *refracts* (cons (cons a b) *refracts*)))
X     ((and (eq act '=>refract) (still-present b))
X        (setq *refracts* (delete (cons a b) *refracts*))
X        (setq rate (rating-part (get a 'topnode)))
X        (removecs a b)
X        (insertcs a b rate))
X     (t (!%warn "back: cannot undo action" (list act a))))
X      (go top)
Xfin   (setq *recording* save)
X      (setq *refracts* nil)
X      (return nil)))
```

```
X
X% still-present makes sure that the user has not deleted something
X% from wm which occurs in the instantiation about to be restored; it
X% makes the check by determining whether each wme still has a time tag.
X(de still-present (data)
X   (prog nil
Xl:    (cond
X     ((atom data) (return t))
X     ((creation-time (car data)) (setq data (cdr data)) (go l:))
X     (t (return nil)))) )
X
X(de back-print (x)
X   (prog (port)
X      (setq port (trace-file))
X      (terpri port)
X      (print x port)))
X
X%%% Functions to show how close rules are to firing
X
X(df matches rule-list
X   (!!mapc (function matches2) rule-list)
X   (terpri))
X
X(de matches2 (p)
X   (cond
X      ((atom p)
X     (terpri)
X     (terpri)
X     (princ p)
X     (matches3 (get p 'backpointers) 2 (ncons 1)))) )
X
X(de matches3 (nodes ce part)
X   (cond
X      ((not (null nodes))
X     (terpri)
X     (princ " ** matches for ")
X     (princ part)
X     (princ " ** ")
X     (!!mapc (function write-elms) (find-left-mem (car nodes)))
X     (terpri)
X     (princ " ** matches for ")
X     (princ (ncons ce))
X     (princ " ** ")
X     (!!mapc (function write-elms) (find-right-mem (car nodes)))
X     (matches3 (cdr nodes) (iadd1 ce) (cons ce part)))) )
X
X(de write-elms (wme-or-count)
X   (cond
X      ((pairp wme-or-count)
X     (terpri)
X     (!!mapc (function write-elms2) wme-or-count))))
X
X(de write-elms2 (x) (princ "  ") (princ (creation-time x)))
```

```
X
X(de find-left-mem (node)
X   (cond
X      ((eq (car node) '&and) (memory-part (caddr node)))
X      (t (car (caddr node)))) )
X
X(de find-right-mem (node) (memory-part (cadddr node)))
X
X%%% Check the RHSs of productions
X
X(de check-rhs (rhs) (!!mapc (function check-action) rhs))
X
X(de check-action (x)
X   (prog (a)
X      (cond ((atom x) (!%warn "atomic action" x) (return nil)))
X      (setq a (setq *action-type* (car x)))
X      (cond
X     ((eq a 'bind) (check-bind x))
X     ((eq a 'cbind) (check-cbind x))
X     ((eq a 'make) (check-make x))
X     ((eq a 'modify) (check-modify x))
X     ((eq a 'remove) (check-remove x))
X     ((eq a 'write) (check-write x))
X     ((eq a 'call) (check-call x))
X     ((eq a 'halt) (check-halt x))
X     ((eq a 'openfile) (check-openfile x))
X     ((eq a 'closefile) (check-closefile x))
X     ((eq a 'default) (check-default x))
X     ((eq a 'build) (check-build x))
X     (t (!%warn "undefined rhs action" a)))) )
X
X(de check-build (z)
X   (and (null (cdr z)) (!%warn "needs arguments" z))
X   (check-build-collect (cdr z)))
X
X(de check-build-collect (args)
X   (prog (r)
Xtop   (cond ((null args) (return nil)))
X      (setq r (car args))
X      (setq args (cdr args))
X      (cond
X     ((pairp r) (check-build-collect r))
X     ((eq r '!!)
X        (and (null args) (!%warn "nothing to evaluate" r))
X        (check-rhs-value (car args))
X        (setq args (cdr args))))
X      (go top)))
X
X(de check-remove (z)
X   (and (null (cdr z)) (!%warn "needs arguments" z))
X   (!!mapc (function check-rhs-ce-var) (cdr z)))
X
X(de check-make (z)
```

```
X   (and (null (cdr z)) (!%warn "needs arguments" z))
X   (check-change& (cdr z)))
X
X(de check-openfile (z)
X   (and (null (cdr z)) (!%warn "needs arguments" z))
X   (check-change& (cdr z)))
X
X(de check-closefile (z)
X   (and (null (cdr z)) (!%warn "needs arguments" z))
X   (check-change& (cdr z)))
X
X(de check-default (z)
X   (and (null (cdr z)) (!%warn "needs arguments" z))
X   (check-change& (cdr z)))
X
X(de check-modify (z)
X   (and (null (cdr z)) (!%warn "needs arguments" z))
X   (check-rhs-ce-var (cadr z))
X   (and (null (cddr z)) (!%warn "no changes to make" z))
X   (check-change& (cddr z)))
X
X(de check-write (z)
X   (and (null (cdr z)) (!%warn "needs arguments" z))
X   (check-change& (cdr z)))
X
X(de check-call (z)
X   (prog (f)
X      (and (null (cdr z)) (!%warn "needs arguments" z))
X      (setq f (cadr z))
X      (and
X     (variablep f)
X     (!%warn "function name must be a constant" z))
X      (or
X     (idp f)
X     (!%warn "function name must be a symbolic atom" f))
X      (or
X     (externalp f)
X     (!%warn "function name not declared external" f))
X      (check-change& (cddr z))))
X
X(de check-halt (z)
X   (or (null (cdr z)) (!%warn "does not take arguments" z)))
X
X(de check-cbind (z)
X   (prog (v)
X      (or (eq (length z) 2) (!%warn "takes only one argument" z))
X      (setq v (cadr z))
X      (or (variablep v) (!%warn "takes variable as argument" z))
X      (note-ce-variable v)))
X
X(de check-bind (z)
X   (prog (v)
X      (or (igreaterp (length z) 1) (!%warn "needs arguments" z))
```

```
X      (setq v (cadr z))
X      (or (variablep v) (!%warn "takes variable as argument" z))
X      (note-variable v)
X      (check-change& (cddr z))))
X
X(de check-change& (z)
X   (prog (r tab-flag)
X      (setq tab-flag nil)
Xla    (cond ((atom z) (return nil)))
X      (setq r (car z))
X      (setq z (cdr z))
X      (cond
X     ((eq r '!^)
X        (and
X           tab-flag
X           (!%warn "no value before this tab" (car z)))
X        (setq tab-flag t)
X        (check-tab-index (car z))
X        (setq z (cdr z)))
X     ((eq r '!/) (setq tab-flag nil) (setq z (cdr z)))
X     (t (setq tab-flag nil) (check-rhs-value r)))
X      (go la)))
X
X(de check-rhs-ce-var (v)
X   (cond
X      ((and (not (numberp v)) (not (ce-bound? v)))
X     (!%warn "unbound element variable" v))
X      ((and (numberp v) (or (lessp v 1) (greaterp v *ce-count*)))
X     (!%warn "numeric element designator out of bounds" v))))
X
X(de check-rhs-value (x)
X   (cond ((pairp x) (check-rhs-function x)) (t (check-rhs-atomic x))))
X
X(de check-rhs-atomic (x)
X   (and
X      (variablep x)
X      (not (bound? x))
X      (!%warn "unbound variable" x)))
X
X(de check-rhs-function (x)
X   (prog (a)
X      (setq a (car x))
X      (cond
X     ((eq a 'compute) (check-compute x))
X     ((eq a 'arith) (check-compute x))
X     ((eq a 'substr) (check-substr x))
X     ((eq a 'accept) (check-accept x))
X     ((eq a 'acceptline) (check-acceptline x))
X     ((eq a 'crlf) (check-crlf x))
X     ((eq a 'genatom) (check-genatom x))
X     ((eq a 'litval) (check-litval x))
X     ((eq a 'tabto) (check-tabto x))
X     ((eq a 'rjust) (check-rjust x))
```

```
X     ((not (externalp a))
X        (!%warn "rhs function not declared external" a)))  )
X
X(de check-litval (x)
X   (or (eq (length x) 2) (!%warn "wrong number of arguments" x))
X   (check-rhs-atomic (cadr x)))
X
X(de check-accept (x)
X   (cond
X      ((eq (length x) 1) nil)
X      ((eq (length x) 2) (check-rhs-atomic (cadr x)))
X      (t (!%warn "too many arguments" x))))
X
X(de check-acceptline (x)
X   (!!mapc (function check-rhs-atomic) (cdr x)))
X
X(de check-crlf (x) (check-0-args x))
X
X(de check-genatom (x) (check-0-args x))
X
X(de check-tabto (x)
X   (or (eq (length x) 2) (!%warn "wrong number of arguments" x))
X   (check-print-control (cadr x)))
X
X(de check-rjust (x)
X   (or (eq (length x) 2) (!%warn "wrong number of arguments" x))
X   (check-print-control (cadr x)))
X
X(de check-0-args (x)
X   (or (eq (length x) 1) (!%warn "should not have arguments" x)))
X
X(de check-substr (x)
X   (or (eq (length x) 4) (!%warn "wrong number of arguments" x))
X   (check-rhs-ce-var (cadr x))
X   (check-substr-index (caddr x))
X   (check-last-substr-index (cadddr x)))
X
X(de check-compute (x) (check-arithmetic (cdr x)))
X
X(de check-arithmetic (l)
X   (cond
X      ((atom l) (!%warn "syntax error in arithmetic expression" l))
X      ((atom (cdr l)) (check-term (car l)))
X      ((not (memq (cadr l) '(+ - * !/)))
X     (!%warn "unknown operator" l))
X      (t (check-term (car l)) (check-arithmetic (cddr l))))  )
X
X(de check-term (x)
X   (cond ((pairp x) (check-arithmetic x)) (t (check-rhs-atomic x))))
X
X(de check-last-substr-index (x)
X   (or (eq x 'inf) (check-substr-index x)))
X
```

```
X(de check-substr-index (x)
X   (prog (v)
X      (cond ((bound? x) (return x)))
X      (setq v ($litbind x))
X      (cond
X     ((not (numberp v))
X        (!%warn "unbound symbol used as index in substr" x))
X     ((or (lessp v 1) (greaterp v 127))
X        (!%warn "index out of bounds in tab" x)))) )
X
X(de check-print-control (x)
X   (prog ()
X      (cond ((bound? x) (return x)))
X      (cond
X     ((or (not (numberp x)) (lessp x 1) (greaterp x 127))
X        (!%warn "illegal value for printer control" x)))) )
X
X(de check-tab-index (x)
X   (prog (v)
X      (cond ((bound? x) (return x)))
X      (setq v ($litbind x))
X      (cond
X     ((not (numberp v))
X        (!%warn "unbound symbol occurs after ^" x))
X     ((or (lessp v 1) (greaterp v 127))
X        (!%warn "index out of bounds after ^" x)))) )
X
X(de note-variable (var)
X   (setq *rhs-bound-vars* (cons var *rhs-bound-vars*)))
X
X(de bound? (var) (or (memq var *rhs-bound-vars*) (var-dope var)))
X
X(de note-ce-variable (ce-var)
X   (setq *rhs-bound-ce-vars* (cons ce-var *rhs-bound-ce-vars*)))
X
X(de ce-bound? (ce-var)
X   (or (memq ce-var *rhs-bound-ce-vars*) (ce-var-dope ce-var)))
X
X%%% Top level routines
X
X(de process-changes (adds dels)
X   (prog (x)
Xprocess-deletes
X      (cond ((atom dels) (go process-adds)))
X      (setq x (car dels))
X      (setq dels (cdr dels))
X      (remove-from-wm x)
X      (go process-deletes)
Xprocess-adds
X      (cond ((atom adds) (return nil)))
X      (setq x (car adds))
X      (setq adds (cdr adds))
X      (add-to-wm x nil)
```

```
X      (go process-adds)))
X
X(de main nil
X   (prog (instance r)
X      (setq *halt-flag* nil)
X      (setq *break-flag* nil)
X      (setq instance nil)
Xdil   (setq *phase* 'conflict-resolution)
X      (cond
X     (*halt-flag* (setq r "end -- explicit halt") (go finis))
X     ((izerop *remaining-cycles*)
X        (setq r '***break***)
X        (setq *break-flag* t)
X        (go finis))
X     (*break-flag* (setq r '***break***) (go finis)))
X      (setq *remaining-cycles* (isub1 *remaining-cycles*))
X      (setq instance (conflict-resolution))
X      (cond
X     ((not instance)
X        (setq r "end -- no production true")
X        (go finis)))
X      (setq *phase* (car instance))
X      (accum-stats)
X      (eval-rhs (car instance) (cdr instance))
X      (check-limits)
X      (and (broken (car instance)) (setq *break-flag* t))
X      (go dil)
Xfinis (setq *p-name* nil)
X      (return r)))
X
X(de do-continue (wmi)
X   (cond
X      (*critical*
X     (terpri)
X     (princ "warning: network may be inconsistent")))
X   (process-changes wmi nil)
X   (print-times (main)))
X
X(de accum-stats nil
X   (setq *cycle-count* (iadd1 *cycle-count*))
X   (setq *total-token* (iplus *total-token* *current-token*))
X   (cond
X      ((igreaterp *current-token* *max-token*)
X     (setq *max-token* *current-token*)))
X   (setq *total-wm* (iplus *total-wm* *current-wm*))
X   (cond
X      ((greaterp *current-wm* *max-wm*)
X     (setq *max-wm* *current-wm*))))
X
X(de print-times (mess)
X   (prog (cc ac)
X      (cond (*break-flag* (terpri) (return mess)))
X      (setq cc (plus (float *cycle-count*) 10.0e-20))
```

```
X      (setq ac (plus (float *action-count*) 1.0e-20))
X      (terpri)
X      (princ mess)
X      (pm-size)
X      (printlinec
X     (list
X        *cycle-count*
X        'firings
X        (list *action-count* 'rhs 'actions)))
X      (terpri)
X      (printlinec
X     (list
X        (round (quotient (float *total-wm*) cc))
X        'mean
X        'working
X        'memory
X        'size
X        (list *max-wm* 'maximum)))
X      (terpri)
X      (printlinec
X     (list
X        (round (quotient (float *total-cs*) cc))
X        'mean
X        'conflict
X        'set
X        'size
X        (list *max-cs* 'maximum)))
X      (terpri)
X      (printlinec
X     (list
X        (round (quotient (float *total-token*) cc))
X        'mean
X        'token
X        'memory
X        'size
X        (list *max-token* 'maximum)))
X      (terpri)))
X
X(de pm-size nil
X   (terpri)
X   (printlinec
X      (list
X     *pcount*
X     'productions
X     (list *real-cnt* '!/ *virtual-cnt* 'nodes)))
X   (terpri))
X
X(de check-limits nil
X   (cond
X      ((igreaterp (length *conflict-set*) *limit-cs*)
X     (terpri)
X     (terpri)
X     (printlinec
```

```
X       (list
X          "conflict set size exceeded the limit of"
X          *limit-cs*
X          "after"
X          *p-name*))
X     (setq *halt-flag* t)))
X   (cond
X      ((igreaterp *current-token* *limit-token*)
X     (terpri)
X     (terpri)
X     (printlinec
X        (list
X           "token memory size exceeded the limit of"
X           *limit-token*
X           "after"
X           *p-name*))
X     (setq *halt-flag* t))))
X
X(de top-level-remove (z)
X   (cond
X      ((equal z '(*)) (process-changes nil (get-wm nil)))
X      (t (process-changes nil (get-wm z)))) )
X
X(df excise z (!!mapc (function excise-p) z))
X
X(df run z
X   (cond
X      ((atom z) (setq *remaining-cycles* 1000000) (do-continue nil))
X      ((and (atom (cdr z)) (numberp (car z)) (greaterp (car z) 0))
X     (setq *remaining-cycles* (car z))
X     (do-continue nil))
X      (t 'what?)))
X
X(df strategy z
X   (cond
X      ((atom z) *strategy*)
X      ((equal z '(lex)) (setq *strategy* 'lex))
X      ((equal z '(mea)) (setq *strategy* 'mea))
X      (t 'what?)))
X
X(df cs z (cond ((atom z) (conflict-set)) (t 'what?)))
X
X(df watch z
X   (cond
X      ((equal z '(0)) (setq *wtrace* nil) (setq *ptrace* nil) 0)
X      ((equal z '(1)) (setq *wtrace* nil) (setq *ptrace* t) 1)
X      ((equal z '(2)) (setq *wtrace* t) (setq *ptrace* t) 2)
X      ((equal z '(3))
X     (setq *wtrace* t)
X     (setq *ptrace* t)
X     '(2 -- conflict set trace not supported))
X      ((and (atom z) (null *ptrace*)) 0)
X      ((and (atom z) (null *wtrace*)) 1)
```

```
X      ((atom z) 2)
X      (t 'what?)))
X
X(df external z (catch !!error!! (external2 z)))
X
X(de external2 (z) (!!mapc (function external3) z))
X
X(de external3 (x)
X   (cond
X      ((idp x) (putprop x t 'external-routine))
X      (t (!%error "not a legal function name" x))))
X
X(de externalp (x)
X   (cond
X      ((idp x) (get x 'external-routine))
X      (t (!%warn "not a legal function name" x) nil)))
X
X(df pbreak z
X   (cond
X      ((atom z) (terpri) *brkpts*)
X      (t (!!mapc (function pbreak2) z) nil)))
X
X(de pbreak2 (rule)
X   (cond
X      ((not (idp rule)) (!%warn "illegal name" rule))
X      ((not (get rule 'topnode)) (!%warn "not a production" rule))
X      ((memq rule *brkpts*) (setq *brkpts* (rematm rule *brkpts*)))
X      (t (setq *brkpts* (cons rule *brkpts*)))) )
X
X(de rematm (atm list)
X   (cond
X      ((atom list) list)
X      ((eq atm (car list)) (rematm atm (cdr list)))
X      (t (cons (car list) (rematm atm (cdr list)))) ))
X
X(de broken (rule) (memq rule *brkpts*))
X
X(i-g-v)
X
X(setsyntax '!{ 'read!-macro nil)
X(setsyntax "{}" 'letter t)
X(setsyntax "{}" 'break-character nil)
X(car!-nil!-legal t)
X
X
Xfin
X
SHAR_EOF
chmod 0600 ops5 || echo "restore of ops5 fails"
rm -f s2_seq_.tmp
echo "You have unpacked the last part"
exit 0
```

From 1202255498111003920
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,e8e66e6b0d5c7d5e,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-06-09 20:48:15 PST
Path: utzoo!attcan!uunet!lll-winken!lll-tis!helios.ee.lbl.gov!pasteur!agate!
saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i052: assist2 -- Desk accessory utility part01/02
Keywords: shar, Alcyon, Laser
Message-ID: <3695@saturn.ucsc.edu>
Date: 10 Jun 88 03:48:15 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 901
Approved: koreth@ssyx.ucsc.edu

Submitted-by: cst.prl.philips.nl!meulenbr (Frans Meulenbroeks)
Posting-number: Volume 1, Issue 52
Archive-name: assist2/part01

[This accessory contains a clock-setter and some disk utilities. Uuencoded
 resource files are included. I got it to compile under Laser C. -sg]

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
# This is part 1 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
#
#    Run the following text with /bin/sh to create:
#      ASSIST2.C
#      ASSIST2.H
#      ASSIST2.MAK
#      ASSIST2.RSC
#      ASSIST2.RSD
#      INCLUDE.H
#
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
sed 's/^X//' << 'SHAR_EOF' > ASSIST2.C &&
X#include "assist2.h"
X#include "include.h"
X#include "gemdefs.h"
X#include "gembind.h"
X
X#define FILENAME     "ASSIST2.RSC"
X#define ACCNAME          "  The Assistant "
X
X
```

```
X/*****************************************************************/
X/***     If you have a battery backup of your systemclock      ***/
X/***  please tell the compiler witch timer keeps on running    ***/
X/***                   >>> System clock (GEM)                  ***/
X/***             >>> Keyboard processor clock (KEY)            ***/
X/*****************************************************************/
X
X#define GEM                0
X
X/*****************************************************************/
X/***       In Europa there is another date-format then         ***/
X/***                    there is in the USA                    ***/
X/***                define the date-format here               ***/
X/***                    >>> 31-12-87  (EUR)                    ***/
X/***                    >>> 12-31-87  (USA)                    ***/
X/*****************************************************************/
X
X#define EUR                0
X
X/*****************************************************************/
X/***                Decaration of the variables                ***/
X/*****************************************************************/
X
XOBJECT          *tree;
Xint       contrl[12],intin[128],intout[128],ptsin[128],ptsout[128];
Xint       in[11],out[57];
Xextern    int gl_apid;
Xshort     btime=TRUE,bdate=FALSE,balarm=FALSE;
Xchar      td_string[18] = "00.00.00 00:00:00";
Xchar      *time_string="000000",*date_string="000000";
Xchar      wait[] = { 0xff, 20 ,0xff, 0 };
Xint       first,vn,zh,button,wh;
Xint               bx,by,bw,bh,xdesk,ydesk,wdesk,hdesk;
X
X
Xint info_track;
Xint info_sector;
Xint info_side;
Xlong info_free;
Xchar info_name[14];
X
X
X
Xmain()
X{
Xregister  long    time;    /*** use register for speed ***/
Xregister  char    temp;
Xint       event,h;
X
Xint eind;
Xchar bfree[7];
XOBJECT  *tree;
Xappl_init();
```

```
Xfirst=TRUE;
X                   /*****************************************/
X                   /*** Copy the time and date              ***/
X                   /*** From GEM (system) to BIOS (keyboard) ***/
X                   /*** or                                  ***/
X                   /*** From BIOS (keyboard) to GEM (system) ***/
X                   /*****************************************/
X/*
X#ifdef GEM
Xtel_bios(Tgetdate(),Tgettime());
X#else
Xtel_gem(Gettime());
X#endif
X
X*/
Xif(!rsrc_load(FILENAME))   /*** try to load the RSC-file ***/
X        {
X        form_alert(1,"[3][I'm sorry, but I can't find|the RESOURCE-file.][ #&%
$*##&^ ]");
X        appl_exit();
X        }
Xelse
X        {
X        vn = graf_handle(&h,&zh,&h,&h);  zh -= 2;
X        v_opnvwk(&h,&vn,out);
X        vst_color(vn,1);
X    wind_get(0,WF_WXYWH,&xdesk,&ydesk,&wdesk,&hdesk);
X                /*** Put program-name into the desktop  ***/
X
X        menu_register(gl_apid,ACCNAME);
X        rsrc_gaddr(0,CLOCK,&tree);
X        set_text(tree,TIME,time_string);
X        set_text(tree,DATE,date_string);
X        while(TRUE)                  /*** means always ***/
X                {
X                time = Gettime();       /*** get the time and date ***/
X                            /*** from the keyboard ***/
X
X                            /*get the second's*/
X                temp = ((((char)time) & 0x1f)*2);
X                convert_time(temp,15,4);
X
X                            /*get the minutes*/
X                time >>= 5;
X                temp = ((char)time) & 0x3f;
X                convert_time(temp,12,2);
X
X                            /*get the hours*/
X                time >>=6;
X                temp = time & 0x1f;
X                convert_time(temp,9,0);
X
X                            /*get the day*/
```

```
X                time >>=5;
X                temp = ((char)time) & 0x1f;
X                #ifdef EUR
X                convert_date(temp,0,0); /*Europe format*/
X                #else
X                convert_date(temp,3,2); /*USA format*/
X                #endif
X
X                                /*get the month*/
X                time >>=5;
X                temp = ((char)time) & 0x0f;
X                #ifdef EUR
X                convert_date(temp,3,2);
X                #else
X                convert_date(temp,0,0);
X                #endif
X
X                                /*get the year*/
X                temp = ((char)((time >> 4) + 80L) % 100) & 0x7f;
X                convert_date(temp,6,4);
X
X                rsrc_gaddr(0,CLOCK,&tree);
X
X                    /*****************************************/
X                    /*** if alarm is set then               ***/
X                    /*** check if the timestring is the same  ***/
X                    /*** as the alarmstring.                ***/
X                    /*****************************************/
X
X                if (balarm==TRUE && (strncmp(( (TEDINFO *) (tree + ALARM)-
>ob_spec )->te_ptext,time_string,4))==0)
X                        {
X                        balarm=FALSE;        /*** set alarm off ***/
X                        setp(tree,ALARMON,NORMAL);
X                        setp(tree,ALARMOFF,SELECTED);
X                        ping();              /*** give some sound ***/
X                        do_dialog(ALERT1,0); /*** tel the user it's time ***/
X                        }
X                if (btime == TRUE && bdate == TRUE)
X                        {
X                             /*** display the      ***/
X                             /*** whole timestring ***/
X
X                        v_gtext(vn,502,zh,td_string);
X                        }
X                else
X                        {
X                        if (btime == TRUE &&  bdate == FALSE)
X                                {
X                             /*** display only ***/
X                             /*** the date     ***/
X
X                                v_gtext(vn,574,zh,&td_string[9]);
```

```
X                               }
X                       else
X                               {
X                               if (btime == FALSE && bdate == TRUE)
X                                       {
X                                    /*** make an early end to   ***/
X                                    /*** the string and display ***/
X                                    /*** the time              ***/
X
X                                       td_string[9]=0;
X                                       v_gtext(vn,574,zh,td_string);
X                                       }
X                               }
X                       }
X               if (((date_string[5]-'0')<7)&&( first==TRUE))
X                       {
X                       ping();
X                       first=FALSE;
X                       wnd_opn();
X                       clock();
X                       wnd_cls();
X                       }
X
X                   /*****************************************/
X                   /*** check wat's happening.            ***/
X                   /*****************************************/
X
X               event = evnt_multi(MU_TIMER | MU_MESAG,0,0,0,0,0,0,0,0,0,0,0,
X                                  0,0,out,2000,0,&h,&h,&h,&h,&h,&h);
X               if((event & MU_MESAG) && (*out == AC_OPEN))
X                       {
X                       wnd_opn();
X                       rsrc_gaddr(0,MAIN,&tree);
X                       eind = FALSE;
X                       while(eind != TRUE)
X                               {
X                               ltoa(Malloc(-1L),bfree);
X                               set_text(tree,MBFREE,bfree);
X                               form_center(tree,&bx,&by,&bw,&bh);
X                               form_dial(0,0,0,0,0,bx,by,bw,bh);
X                               form_dial(1,0,0,0,0,bx,by,bw,bh);
X                               objc_draw(tree,0,32,bx,by,bw,bh);
X                               button=form_do(tree,0);
X                               objc_change(tree,button,0,bx,by,bw,bh,NORMAL,0);
X                               form_dial(3,0,0,0,0,bx,by,bw,bh);   /*** erase
dialog ***/
X                               switch(button)
X                                       {
X
X                                       case ABOUT:
X                                       do_dialog(INTRO,0);
X                                       break;
X
```

```
X                                       case MCLOCK:
X                                       clock();
X                                       break;
X
X                                       case MFORMAT:
X                                       format();
X                                       break;
X
X                                       case MAKEDIR:
X                                       makedir();
X                                       break;
X
X                                       case MINFOA:
X                                       case MINFOB:
X                                       disk_info(button-MINFOA);
X                                       break;
X
X                                       case MERASEA:
X                                       case MERASEB:
X                                       disk_erase(button-MERASEA);
X                                       break;
X
X                                       case MCANCEL:
X                                       eind=TRUE;
X                                       break;
X                                       }  /* end of switch */
X                                  }  /* end of while */
X                             wnd_cls();
X                           }
X                  }
X         appl_exit();
X         }
X}
X
X
X
Xdisk_erase(drive)
Xint drive;
X{
Xif (insert(drive)==OKINSERT)
X         {
X         if (get_info(drive)==TRUE)
X                  {
X                  make_boot(info_track,info_sector,info_side-1,drive,"ASSIST2
FMT");
X                  disk_info(drive);
X                  }
X         }
X}
X
X
X
Xdisk_info(drive)
```

```
Xint drive;
X{
XOBJECT  *tree;
Xint again=TRUE;
Xchar    *track="00",*sector="00",*side="0",*free="000000";
X
Xrsrc_gaddr(0,DISKINFO,&tree);
Xwhile (again == TRUE)
X        {
X        Fsfirst("$%&&&&.&^&",255);
X        again=FALSE;
X        info_name[0]=0;
X        if (get_info(drive)==TRUE)
X                {
X                ltoa((long)info_track,track);
X                ltoa((long)info_sector,sector);
X                ltoa((long)info_side,side);
X                ltoa((long)info_free,free);
X                set_text(tree,INFOTRAC,track);
X                set_text(tree,INFOSECT,sector);
X                set_text(tree,INFOSIDE,side);
X                set_text(tree,INFOFREE,free);
X                set_text(tree,INFONAME,info_name);
X                do_dialog(DISKINFO,0);
X                if (button == INFOAGAI) again=TRUE;
X                }
X        }
X}
X
X
X
Xget_info(drive)
Xint drive;
X{
Xint dummy,error,old_drive;
Xchar  *buffer;
Xlong  buf[4];
XDTABUF *dtabuf;
Xbuffer = malloc(20000);
Xdtabuf = (DTABUF *) Fgetdta();
Xold_drive=Dgetdrv();
XDsetdrv(drive);
Xdummy=Floprd(buffer,0L,drive,1,0,0,1);
Xif(dummy < 0)
X    {
X    do_dialog(DISKERRO,0);
X    error=FALSE;
X    }
Xelse
X    {
X    if (Fsfirst("*.*",8)==0)
X        {
X        for(dummy=0; dummy<14; dummy++) info_name[dummy]=dtabuf->name[dummy];
```

```
X        info_name[dummy++]=0;
X        }
X        Dfree(&buf,drive+1);
X        info_free=buf[0]*buf[2]*buf[3];
X        info_side=get_no(0x1a,buffer);
X        info_track=get_no(0x13,buffer)/(get_no(0x18,buffer) *
get_no(0x1a,buffer));
X        info_sector=get_no(0x18,buffer);
X        error=TRUE;
X        }
Xfree(buffer);
XDsetdrv(old_drive);
Xreturn(error);
X}
X
X
X
X
X/*****************************************************************/
X/***                 convert the integer time                  ***/
X/***                to ASCII and put that into                 ***/
X/***                      two strings.                         ***/
X/*****************************************************************/
Xconvert_time(time,n1,n2)
Xregister char time;
Xint       n1,n2;
X{
Xtd_string[n1] = time_string[n2] = time /10 +'0';
Xtd_string[n1+1] = time_string[n2+1] = time %10 +'0';
X}
X
X
X/*****************************************************************/
X/***                 convert the integer date                  ***/
X/***                to ASCII and put that into                 ***/
X/***                      two strings.                         ***/
X/*****************************************************************/
Xconvert_date(date,n1,n2)
Xregister char date;
Xint       n1,n2;
X{
Xtd_string[n1] = date_string[n2] = date / 10 + '0';
Xtd_string[n1+1] = date_string[n2+1] = date % 10 + '0';
X}
X
X
X/*****************************************************************/
X/***          function to let the user set the date,           ***/
X/***          time and alarm and how to display it.            ***/
X/*****************************************************************/
Xclock()
X{
Xchar      date2[6],time2[6];
```

```
Xint       time=0,date=0;
X
X                              /*** copy the date and time     ***/
X                              /*** currently in the dialogbox ***/
Xstrcpy(date2,date_string);
Xstrcpy(time2,time_string);
X                              /*** ask for user respond ***/
Xdo_dialog(CLOCK,TIME);
X                              /*** set some controle var.'s ***/
Xbtime=selectp(tree,TIMEON);
Xbdate=selectp(tree,DATEON);
Xbalarm=selectp(tree,ALARMON);
X                              /*** if exit is SETTIME    ***/
X                              /*** set the time and date ***/
Xif (button == SETTIME)
X     {
X     time |= convert(time_string,0);
X     time <<=6;
X     time |= convert(time_string,2);
X     time <<=5;
X     time |= (convert(time_string,4)/2);
X     Tsettime(time);
X     date |= (convert(date_string,4)-80);
X     date <<=4;
X#ifdef EUR
X     date |= convert(date_string,2);
X     date <<=5;
X     date |= convert(date_string,0);
X#else
X     date |= convert(date_string,0);
X     date <<=5;
X     date |= convert(date_string,2);
X#endif
X     Tsetdate(date);
X     tel_bios(Tgetdate(),Tgettime());
X     }
Xv_gtext(vn,502,zh,"                 "); /***clear the desktop***/
X}
X
X
X
X/*****************************************************************/
X/***    function to convert a 2byte-string into an integer    ***/
X/*****************************************************************/
Xconvert(string,number)
Xchar *string;
Xint number;
X{
Xreturn((string[number]-'0')*10 +(string[number+1]-'0'));
X}
X
X
X
```

```
X
X/*****************************************************************/
X/***               function to give some sound                 ***/
X/*****************************************************************/
Xping()
X{
XCconout(7);
Xwhile(Dosound(-1L));
XDosound(wait);
Xwhile(Dosound(-1L));
XCconout(7);
X}
X
X
X/*****************************************************************/
X/***function to get the time from GEM and send it to the bios ***/
X/*****************************************************************/
Xtel_bios(timedate)
Xlong      timedate;
X{
XSettime(timedate);
X}
X
X
X/*****************************************************************/
X/***                    reverse of the above                   ***/
X/*****************************************************************/
Xtel_gem(date,time)
Xint date,time;
X{
XTsetdate(date);
XTsettime(time);
X}
X
X
X
X/*****************************************************************/
X/***          function to set the state of an object           ***/
X/***                to NORMAL,SELECTED,......                  ***/
X/*****************************************************************/
X
Xsetp(tree,object,bit)
XOBJECT *tree;
Xint         object,bit;
X{
Xobjc_change(tree,object,0,bx,by,bw,bh,bit,0);
X}
X
X
X
Xdo_format(tr,se,si,dr,diskn)
Xint tr,se,si,dr;
Xchar    *diskn;
```

```
X{
XOBJECT  *tree;
Xint       dt,button;
Xchar      trs[2],sis[2],*buffer;
Xsis[0]= '0';
Xbuffer = malloc(20000);
Xrsrc_gaddr(0,GO,&tree);
X                              /*** set walking-bar to start ***/
Xinit_prog();
X                              /*** let some RSC-strings point ***/
X                              /*** to our own strings         ***/
Xset_text(tree,GOTRACK,trs);
Xset_text(tree,GOSIDE,sis);
X                              /*** display a dialog box ***/
Xform_center(tree,&bx,&by,&bw,&bh);      /*** get coordinates ***/
Xform_dial(0,0,0,0,0,bx,by,bw,bh );  /*** reserve screen  ***/
Xform_dial(1,0,0,0,0,bx,by,bw,bh);       /*** make growbox    ***/
Xobjc_draw(tree,0,MAX_DEPTH,bx,by,bw,bh);
Xfor(dt=0;dt<tr;dt++)
X     {
X                              /*** let the user see ***/
X                              /*** what's going on  ***/
X     set_prog((float)dt,(float)tr-1,bx,by,bw,bh);
X     ltoa((long)dt,trs);
X     objc_draw(tree,GOTRACK,0,bx,by,bw,bh);
X     objc_draw(tree,GOSIDE,0,bx,by,bw,bh);
X                              /*** and format side one ***/
X     button = Flopfmt(buffer,0L,dr,se,dt,0,01,0x87654321L,0xe5e5);
X                              /*** if asked for.   ***/
X                              /*** format side two ***/
X     if((si == 1) && (button == 0))
X          {
X          sis[0] +=1;
X          objc_draw(tree,GOSIDE,0,bx,by,bw,bh);
X          sis[0] -=1;
X          button = Flopfmt(buffer,0L,dr,se,dt,1,01,0x87654321L,0xe5e5);
X          }
X                              /*** if something goes wrong ***/
X                              /*** button is true           ***/
X     if(button != 0) dt = (tr+1);
X     }
Xform_dial(3,0,0,0,0,bx,by,bw,bh);   /*** erase dialog ***/
Xfree(buffer);
Xif (button == FALSE) make_boot(tr,se,si,dr,diskn);
Xelse do_dialog(DISKERRO,0);
X}
X
X
X
X
X
Xformat()
X{
```

```
X
XOBJECT          *tree;
Xint     sectors,tracks=80,sides,drive,eind;
Xchar    *diskname="EXTENDEDFMT",*track="80";
X
Xrsrc_gaddr(0,EXFORMAT,&tree);
Xset_text(tree,DISKNAME,diskname);
Xset_text(tree,TRACKNUM,track);
Xform_center(tree,&bx,&by,&bw,&bh);
Xform_dial(0,0,0,0,0,bx,by,bw,bh);
Xform_dial(1,0,0,0,0,bx,by,bw,bh);
Xobjc_draw(tree,0,32,bx,by,bw,bh);
Xeind = FALSE;
Xwhile(eind != TRUE)
X        {
X        button=form_do(tree,DISKNAME);
X        objc_change(tree,button,0,bx,by,bw,bh,NORMAL,0);
X        switch(button){
X
X                case FORMAT:
X                sides= selectp(tree,DS);
X                drive= selectp(tree,DRIVEB);
X                sectors=selectp(tree,SEC10)+9;
X                form_dial(3,0,0,0,0,bx,by,bw,bh);   /*** erase dialog ***/
X                                if (insert(drive)==OKINSERT)
X                        {
X                        do_format(tracks,sectors,sides,drive,diskname);
X                        disk_info(drive);
X                        }
X                eind=TRUE;
X                break;
X
X                case CANCEL:
X                eind=TRUE;
X                form_dial(3,0,0,0,0,bx,by,bw,bh);   /*** erase dialog ***/
X                                break;
X
X
X                case TRACKDWN:
X                case TRACKUP:
X                if ((button==TRACKUP)&(tracks<83)) tracks+=1;
X                if ((button==TRACKDWN)&(tracks>80)) tracks-=1;
X                ltoa((long)tracks,track);
X                evnt_timer(100,0);
X                objc_draw(tree,TRACKNUM,0,bx,by,bw,bh);
X                break;
X
X                }
X        }
X}
X
X
X
```

```
X/*****************************************************************/
X/***   function to set the width of the object used by the    ***/
X/***                progress-indicator to zero                 ***/
X/*****************************************************************/
Xinit_prog()
X{
XOBJECT         *tree;
X
Xrsrc_gaddr(0, GO, &tree);
X(tree + BAR)->ob_width = 0;
X}
X
X
Xinsert(drive)
Xint drive;
X{
Xchar *str_drive="X";
XOBJECT  *tree;
Xrsrc_gaddr(0,INSERT,&tree);
Xstr_drive[0]=drive+'A';
Xset_text(tree,AINSERT,str_drive);
Xdo_dialog(INSERT,0);
Xreturn(button);
X}
X
X
Xltoa(number,string)
Xlong number;
Xchar *string;
X{
Xsprintf(string,"%ld  ",number);
X}
X
X
X
X/*****************************************************************/
X/***         function to make an adjusted bootsector          ***/
X/*****************************************************************/
X
Xmake_boot(tr,se,si,dr,diskn)
Xint tr,se,si,dr;
Xchar *diskn;
X{
Xint       total,dummy;
Xchar      *buffer;
Xbuffer=malloc(20000);
X                    /*** Make a new bootsector and ***/
X                    /*** write is to disk          ***/
X
Xtotal=(tr * se);
Xfor(dummy=0; dummy<(se*512); *(buffer + dummy++)=0);
XProtobt(buffer,0x01000000L,(((tr>40)*2)+si),0);
X*(buffer + 0x13)=(total+(si ? total : 0)) &0xff;
```

```
X*(buffer + 0x14)=((total+(si ? total : 0)) >> 8) &0xff;
X*(buffer + 0x18)=se;
XFlopwr(buffer,0L,dr,1,0,0,se);
X
Xfor(dummy=0; dummy<512; buffer[dummy++]=0);
Xfor (dummy=0; dummy<strlen(diskn); dummy++)
X        {
X    diskn[dummy]=toupper(diskn[dummy]);
X    }
Xsprintf(buffer,diskn);
Xbuffer[11]=8;
XFlopwr(buffer,0L,dr,(12-se),(1-si),si,(se-4));
Xfree(buffer);
X}
X
X
X/*****************************************************************/
X/***        function to check if an object is SELECTED        ***/
X/*****************************************************************/
Xselectp(tree,which)
Xlong           tree;
Xint            which;
X{
Xreturn statep(tree,which,SELECTED);
X}
X
X
X
X
X/*****************************************************************/
X/***      function to set the width of an object              ***/
X/***  changing the width of an object and redraw that object  ***/
X/***             wil give us a process-indicator              ***/
X/*****************************************************************/
Xset_prog(value, maxc,x,y,w,h)
Xfloat          value, maxc;
Xint x,y,w,h;
X{
XOBJECT         *tree;
XGRECT          box;
X
Xrsrc_gaddr(0, GO, &tree);
X(tree + BAR)->ob_width=((((tree + BOX)->ob_width ) / maxc)  * value);
Xobjc_draw(tree,BAR,0,x,y,w,h);
X}
X
X
X
Xset_text(tree,object,string)
XOBJECT         *tree;
Xint            object;
Xchar           *string;
X{
```

```
XTEDINFO        *obspec;
X
Xobspec= (TEDINFO *) (tree +object)->ob_spec;
Xobspec->te_ptext=string;
Xobspec->te_txtlen=strlen(string)+1;
X}
X
X
X
X
X
X/*****************************************************************/
X/***          function to get the state of an object          ***/
X/***               to NORMAL,SELECTED,.....                   ***/
X/*****************************************************************/
X
Xstatep(tree,which,bit)
Xlong           tree;
Xint            which,bit;
X{
Xreturn((LWGET(OB_STATE(which)) & bit) != FALSE);
X}
X
X
X/*****************************************************************/
X/***      function to convert an integer in MSDOS-format       ***/
X/***                     to ATARI format                       ***/
X/*****************************************************************/
Xget_no(i,buffer)
Xint i;
Xchar *buffer;
X{
Xint       low,high;
Xlow = *(buffer + i); high = *(buffer+ i+1);
Xif(low < 0) high += 1;
Xreturn(low + (256*high));
X}
X
Xmakedir()
X{
XOBJECT  *tree;
Xchar *path,*name="12345678.123\0";
Xint  dummy;
Xpath = (char *)Malloc(40L);
Xstrcpy(name,"FILENAME.EXT\0");
Xstrcpy(path,"A:\\*.*\0");
Xpath[0]=(Dgetdrv()+'A');
XDsetpath(path);
Xfsel_input(path,name,&button);
Xif (button==1)
X        {
X        for(dummy=strlen(path); dummy>0;dummy--)
X                {
```

```
X                if (path[dummy]=='\\')
X                        {
X                        path[dummy]=0;
X                        dummy=0;
X                        }
X                }
X        Dsetdrv(path[0]-'A');
X        Dsetpath(path);
X        if(Dcreate(name)) do_dialog(DISKERRO,0);
X        }
XMfree(path);
X}
X
X
X
X/*************************************************************/
X/** function to get user-respond from a already drawn dialog.  **/
X/**            And to set the exitbutton to NORMAL.           **/
X/*************************************************************/
X
Xdo_dialog(dialog,edit)
Xint       dialog,edit;
X{
Xrsrc_gaddr(0,dialog,&tree);
Xform_center(tree,&bx,&by,&bw,&bh);      /*** get coordinates ***/
Xform_dial(0,0,0,0,0,bx,by,bw,bh );  /*** reserve screen  ***/
Xform_dial(1,0,0,0,0,bx,by,bw,bh);       /*** make growbox    ***/
Xobjc_draw(tree,0,MAX_DEPTH,bx,by,bw,bh);
Xbutton = form_do(tree,edit);
Xform_dial(2,0,0,0,0,bx,by,bw,bh);       /*** schrink box  ***/
Xform_dial(3,0,0,0,0,bx,by,bw,bh);   /*** erase dialog ***/
Xsetp(tree,button,NORMAL);
X}
X
Xwnd_opn()
X{
Xwh=wind_create(0,xdesk,ydesk,wdesk,hdesk);
Xwind_open(wh,xdesk,ydesk,wdesk,hdesk);
X}
X
Xwnd_cls()
X{
Xwind_close(wh);
Xwind_delete(wh);
X}
X
SHAR_EOF
chmod 0600 ASSIST2.C || echo "restore of ASSIST2.C fails"
sed 's/^X//' << 'SHAR_EOF' > ASSIST2.H &&
X/* resource set indicies for ASSIST2 */
X
X#define EXFORMAT 0     /* form/dialog */
X#define TRACKUP  6     /* BUTTON in tree EXFORMAT */
```

```
X#define TRACKDWN 7    /* BUTTON in tree EXFORMAT */
X#define TRACKNUM 8    /* FTEXT in tree EXFORMAT */
X#define SEC9     12   /* BUTTON in tree EXFORMAT */
X#define SEC10    13   /* BUTTON in tree EXFORMAT */
X#define DISKNAME 14   /* FBOXTEXT in tree EXFORMAT */
X#define FORMAT   15   /* BUTTON in tree EXFORMAT */
X#define CANCEL   16   /* BUTTON in tree EXFORMAT */
X#define SS       20   /* BUTTON in tree EXFORMAT */
X#define DS       21   /* BUTTON in tree EXFORMAT */
X#define DRIVEA   25   /* BUTTON in tree EXFORMAT */
X#define DRIVEB   26   /* BUTTON in tree EXFORMAT */
X
X#define GO       1    /* form/dialog */
X#define GOTRACK  7    /* FTEXT in tree GO */
X#define GOSIDE   8    /* FTEXT in tree GO */
X#define BOX      9    /* TEXT in tree GO */
X#define BAR      10   /* BOXTEXT in tree GO */
X
X#define MAIN     2    /* form/dialog */
X#define MBFREE   3    /* FBOXTEXT in tree MAIN */
X#define MCLOCK   4    /* BUTTON in tree MAIN */
X#define MFORMAT  5    /* BUTTON in tree MAIN */
X#define MCANCEL  6    /* BUTTON in tree MAIN */
X#define MAKEDIR  7    /* BUTTON in tree MAIN */
X#define MINFOA   10   /* BUTTON in tree MAIN */
X#define MINFOB   11   /* BUTTON in tree MAIN */
X#define MERASEA  14   /* BUTTON in tree MAIN */
X#define MERASEB  15   /* BUTTON in tree MAIN */
X#define ABOUT    16   /* BUTTON in tree MAIN */
X
X#define INSERT   3    /* form/dialog */
X#define AINSERT  3    /* FTEXT in tree INSERT */
X#define OKINSERT 5    /* BUTTON in tree INSERT */
X#define NOINSERT 6    /* BUTTON in tree INSERT */
X
X#define DISKERRO 4    /* form/dialog */
X
X#define INTRO    5    /* form/dialog */
X
X#define CLOCK    6    /* form/dialog */
X#define TIME     4    /* FTEXT in tree CLOCK */
X#define TIMEOFF  5    /* BUTTON in tree CLOCK */
X#define TIMEON   6    /* BUTTON in tree CLOCK */
X#define DATE     9    /* FTEXT in tree CLOCK */
X#define DATEOFF  10   /* BUTTON in tree CLOCK */
X#define DATEON   11   /* BUTTON in tree CLOCK */
X#define CLOCKEXI 13   /* BUTTON in tree CLOCK */
X#define ALARM    15   /* FTEXT in tree CLOCK */
X#define ALARMON  16   /* BUTTON in tree CLOCK */
X#define ALARMOFF 18   /* BUTTON in tree CLOCK */
X#define SETTIME  19   /* BUTTON in tree CLOCK */
X
X#define DISKINFO 7    /* form/dialog */
```

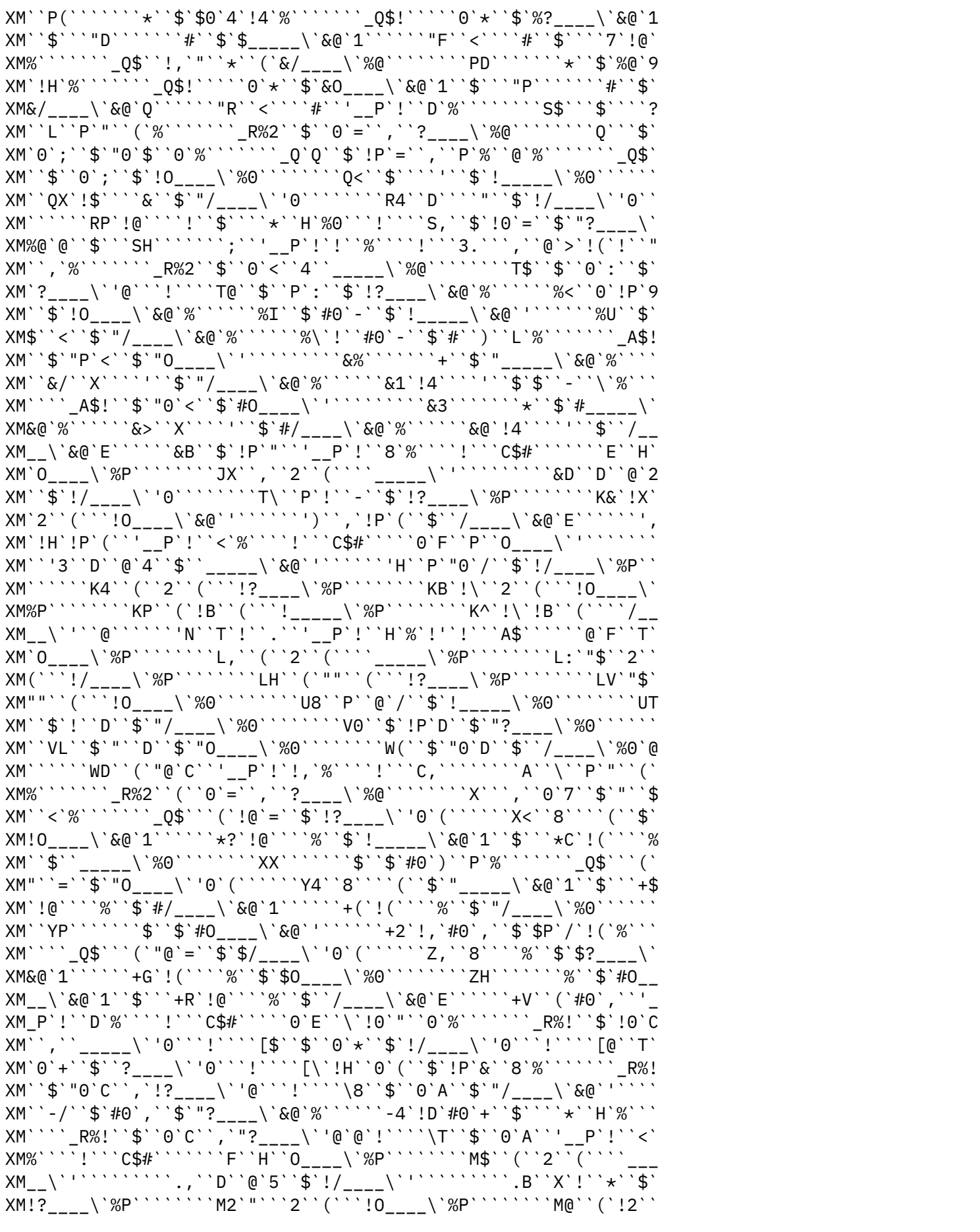
```
X#define INFOTRAC 2    /* FTEXT in tree DISKINFO */
X#define INFOSECT 3    /* FTEXT in tree DISKINFO */
X#define INFOSIDE 4    /* FTEXT in tree DISKINFO */
X#define INFOFREE 6    /* FBOXTEXT in tree DISKINFO */
X#define INFOEXIT 7    /* BUTTON in tree DISKINFO */
X#define INFOAGAI 8    /* BUTTON in tree DISKINFO */
X#define INFONAME 10   /* FBOXTEXT in tree DISKINFO */
X
X#define ALERT1   8    /* form/dialog */
X
XSHAR_EOF
chmod 0600 ASSIST2.H || echo "restore of ASSIST2.H fails"
sed 's/^X//' << 'SHAR_EOF' > ASSIST2.MAK &&
Xkey.c
Xmegamax\mmlink.ttp key.o   megamax\acc.l  -o key.acc
SHAR_EOF
chmod 0600 ASSIST2.MAK || echo "restore of ASSIST2.MAK fails"
sed 's/^X//' << 'SHAR_EOF' > s2_temp_.tmp &&
Xbegin 600 ASSIST2.RSC
XM``$/4`M\#U`*N`JX`"0#N`M\&I``>``)`",````.`````!JT#@\.#R!&3U)-
SHAR_EOF
echo "End of part 1, continue with part 2"
echo "2" > s2_seq_.tmp
exit 0


From -4492480513181833254
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,99dde6561942534,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-06-09 20:48:45 PST
Path: utzoo!attcan!uunet!lll-winken!lll-tis!helios.ee.lbl.gov!pasteur!agate!
saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i053: assist2 -- Desk accessory utility  part02/02
Keywords: shar, Alcyon, Laser
Message-ID: <3696@saturn.ucsc.edu>
Date: 10 Jun 88 03:48:45 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 306
Approved: koreth@ssyx.ucsc.edu

Submitted-by: cst.prl.philips.nl!meulenbr (Frans Meulenbroeks)
Posting-number: Volume 1, Issue 53
Archive-name: assist2/part02

#!/bin/sh
# this is part 2 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file ASSIST2.RSC continued
#
CurArch=2
```

```
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> s2_temp_.tmp
XM051415(@#@\.#P```%1R86-K<P````,`!``X,`!?7P!86`!396-T;W)S````
XM.0`Q,``Q,C,T-38W.#$R,P!$:7-K;F%M92`@7U]?7U]?7U\N7U]?`&YN;FYN
XM;FYN;FYN`$9/4DU!5`!#04Y#14P`4VED97,````Q`#(`1')I=F4```!!`$(`
XM#@\.#R!&3U)-051424Y'(`X/#@\```!4<F%C:R`Z````4VED92`Z````,#``
XM7U\`.3D`,`!?`#D`("`@("`@("`@("`@("`@(````"`@5U))5%1%3B!"62!"
XM15)4(%)/6D5.0D521P````X/#@\@(%1(12!!4U-)4U1!3E0@,B`.#PX/````
XM,#`P,#`P`$)Y=&5S($9R964Z(%]?7U]?7P!86%A86%@`4T-2145.($-,3T-+
XM`$585"X@1D]234%4`$-!3D-%3`!-04M%($1)4@!$25-+($E.1D\`00!"`$1)
XM4TL@15)!4T4`00!"`+T`4&QE87-E(&EN<V5R="!D:7-K`$$`:6YT;R!$<FEV
XM93H@7P!8`$]+`$-!3D-%3`!4:&5R92=S('-O;64@=')O=6)L90!!0D]25`!W
XM:71H('1H92!D:7-K+@!42$4@05-325-404Y4(#(```!7<FET=&5N("@Q.3@W
XM*2!B>3H```!"97)T(%)O>F5N8F5R9P```$MI:VME<G9E96X@,S,Q````,S(P
XM-2!80R`@4W!I:FME;FES<V4```!4:&4@3F5T:&5R;&%N9',````.#PX/(%-#
XM4D5%3BU#3$]#2R`.#PX/````,#`P,#`P`%]?.E]?.E]?`#DY.3DY.0!/1D8`
XM3TX`5&EM90```#`P,#`P,`!?7SI?7SI?7P`Y.3DY.3D`3T9&`$].`$1A=&4`
XM``!%6$E4`#`P,#``7U\Z7U\`.3DY.0!/3@!!;&%R;0```$]&1@!3151424U%
XM`#`P`%1R86-K<SH@7U\`.3D`,#``4V5C=&]R<SH@7U\`.3D`,`!3:61E<SH@
XM7P`Y`#`P,#`P,`!">71E<R!F<F5E.B!?7U]?7U\`.3DY.3DY`$58250`04=!
XM24X`,3(S-#4V-S@N,3(S`$1I<VMN86UE(#H@7U]?7U]?7U]?7U]?`%A86%A8
XM6%A86%A86`!)="=S('1I;64@=&\@<F5M96UB97(`4T]-151(24Y'+@!42$%.
XM2R!93U4`20`````````?\```?_P``.`>```#@`!X`X````\````\````
XM\````<````!````!P````<````!``````````<````/@```!P``````````
XM`/__________SS!A`\\G,\_))S//R2`SS\DG,\_))S//YF<AS__________
XM`````````!_P``!__```X!X``<`.```@#@```#P```#P```#P```!P```
XM`<````!````!P````<````````!P````^````!`````````````  ________
XM___/,&$#SR<SS\DG,\_)(#//R2<SS\DG,\_F9R'/__________\``````!@
XM```#=@``-W<``'=W``!W=P``=W<``'=W``!W=P``?_YX``JN\`!O^<``>^>`
XM`'V_@``^OP```_X```O\```/_```#_P``!_^````````````'GYY^#,8S8PP
XM&,V,,!C-C!X8S?@#&,V``QC-@#,8S8`>&'F```````````````&````-V```W
XM=P``=W<``'=W``!W=P``=W<``'=W``!__G@`>J[P`&_YP`![YX``?;^```#Z_
XM```?_@``"_P```_\```/_```'_X````````````>?GGX,QC-C#`8S8PP&,V,
XM'AC-^`,8S8`#&,V`,QC-@!X8>8````````````8````W8``#=W``!W=P``
XM=W<``'=W``!W=P``=W<``'_^>`!ZKO``;_G```OG@`!]OX``/K\``!_^```+
XM_```#_P```_\```?_@````````!Y^>?@S&,V,,!C-C#`8S8P>&,WX`QC-
XM@`,8S8`S&,V``AAY@````````````!@```#=@``-W<``'=W``!W=P``=W<`
XM`'=W``!W=P``?_YX``JN\`!O^<``>^>```V_@``^OP```_X```O\```/_```
XM#_P``!_^````````````'GYY^#,8S8PP&,V,,!C-C!X8S?@#&,V``QC-@#,8
XMS8`>&'F````````````````````.````#@```X````.````#@``;NX``.[N
XM``#N[@``[NN``!$/P`#_NX``PS,``.^.``#S^```??@``#_P```?X```'^``
XM`````````````````````'#/G[YXV,8P;-C&,&;8QCQCV,8P8=C&,&#/ACX`
XM````````````````X````.````#@```X````.``!N[@``[NX``.[N``#N
XMZX``$0_``/^[@`##,P``[XX``//X``!]^```/_```!_@```?X``````````
XM`````````````<,^?OGC8QC!LV,8P9MC&/&/8QC!AV,8P8,^&/@`````````
```



```
XM(```!_____\`%P`````````MN`"``!2``(````/____\`&@`G``````.M``P`
XI!P`,``$```]0```1V```$N```!1X```5(```%>```!;H```8R```&="M
X`
Xend
SHAR_EOF
uudecode < s2_temp_.tmp && rm -f s2_temp_.tmp &&
chmod 0600 ASSIST2.RSC || echo "restore of ASSIST2.RSC fails"
sed 's/^X//' << 'SHAR_EOF' > ASSIST2.RSD &&
XXXXXSHAR_EOF
chmod 0600 ASSIST2.RSD || echo "restore of ASSIST2.RSD fails"
sed 's/^X//' << 'SHAR_EOF' > INCLUDE.H &&
Xextern  long bios();
Xextern  long xbios();
Xextern  long gemdos();
X
X#define  Floprd(a,b,c,d,e,f,g)  xbios(8,a,b,c,d,e,f,g)
X#define  Flopwr(a,b,c,d,e,f,g)  xbios(9,a,b,c,d,e,f,g)
X#define  Flopfmt(a,b,c,d,e,f,g,h,i)  xbios(10,a,b,c,d,e,f,g,h,i)
X#define  Protobt(a,b,c,d)   xbios(18,a,b,c,d)
X#define  Settime(a)    xbios(22,a)
X#define  Gettime() xbios(23)
X#define  Dosound(a)    xbios(32,a)
X#define Supexec(a)    xbios(38, a)
X
X
X#define  Cconout(a)    gemdos(0x2,a)
X#define  Dsetdrv(a)    gemdos(0x0e,a)
X#define  Dgetdrv() (int)gemdos(0x19)
X#define  Fsetdta(a)    gemdos(0x1a,a)
X#define  Tgetdate()    (int)gemdos(0x2a)
X#define  Tsetdate(a)   gemdos(0x2b,a)
X#define  Tgettime()    (int)gemdos(0x2c)
X#define  Tsettime(a)   gemdos(0x2d,a)
X#define  Fgetdta() gemdos(0x2f)
X#define  Dfree(a,b)    gemdos(0x36,a,b)
X#define  Dcreate(a)    gemdos(0x39,a)
X#define  Dsetpath(a)   gemdos(0x3b,a)
X#define  Dgetpath(a,b) gemdos(0x47,a,b)
X#define  Malloc(a) gemdos(0x48,a)
X#define  Mfree(a)  gemdos(0x49,a)
X#define  Fsfirst(a,b)  gemdos(0x4e,a,b)
X
X
X#define  Mediach(a)    bios(9,a)
X
X
X/*********************************************************/
X
X#define MAX_DEPTH 8        /* max depth of search or draw */
X
X
Xtypedef struct grect
X{
```

```
X    int  g_x;
X    int  g_y;
X    int  g_w;
X    int  g_h;
X} GRECT;
X
X
Xtypedef struct text_edinfo
X{
X    char *te_ptext;         /* ptr to text (must be 1st) */
X    char *te_ptmplt;   /* ptr to template */
X    char *te_pvalid;   /* ptr to validation chrs. */
X    int te_font;       /* font */
X    int te_junk1;      /* junk word */
X    int te_just;       /* justification- left, right... */
X    int te_color;      /* color information word */
X    int te_junk2;      /* junk word */
X    int te_thickness;  /* border thickness    */
X    int te_txtlen;          /* length of text string */
X    int te_tmplen;          /* length of template string */
X} TEDINFO;
X
X
X
Xtypedef struct object
X{
X    int      ob_next;  /* -> object's next sibling */
X    int      ob_head;  /* -> head of object's children */
X    int      ob_tail;  /* -> tail of object's children */
X    unsigned int ob_type;   /* type of object- BOX, CHAR,... */
X    unsigned int ob_flags;  /* flags */
X    unsigned int ob_state;  /* state- SELECTED, OPEN, ... */
X    char *ob_spec;          /* "out"- -> anything else */
X    int      ob_x;          /* upper left corner of object */
X    int      ob_y;          /* upper left corner of object */
X    int      ob_width;/* width of obj    */
X    int      ob_height;     /* height of obj */
X} OBJECT;
X
X
X#define NORMAL    0x0  /* Object states */
X#define SELECTED  0x1
X#define CROSSED   0x2
X#define CHECKED   0x4
X#define DISABLED  0x8
X#define OUTLINED 0x10
X#define SHADOWED 0x20
X
X
X/*************************************************************/
X/***      Some define's missing in the MEGAMAX-compiler       ***/
X/*************************************************************/
X
```

```
X#define TRUE            1
X#define FALSE           0
X#define OB_STATE(x)     (tree + (x) *sizeof(OBJECT)+10)
X#define ADDR            /**/
X#define LBGET(x)        ((char)*((char *)(x)))
X#define LBSET(x,y)      ( *((char *)(x))=y)
X#define LWGET(x)        ((int) *((int *)(x)))
X#define LWSET(x,y)      ( *((int *)(x)) = y)
X#define LLGET(x)        ( *((long *)(x)))
X#define LLSET(x,y)      ( *((long *)(x)) = y)
X#define LBYTE0(x)       ( *((x)+3) )
X#define LBYTE1(x)       ( *((x)+2) )
X#define LBYTE2(x)       ( *((x)+1) )
X#define LBYTE3(x)       ( *(x) )
X
Xextern char *malloc();
X
X
Xtypedef struct dta_buffer
X    {
X        char dummy[21];
X        char file_attr;
X        int  time;
X        int  date;
X        long size;
X        char name[14];
X    } DTABUF;
SHAR_EOF
chmod 0600 INCLUDE.H || echo "restore of INCLUDE.H fails"
rm -f s2_seq_.tmp
echo "You have unpacked the last part"
exit 0
```


From 121664882098858506
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,e0880b930299dc0a,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-06-14 09:05:30 PST
Path: utzoo!attcan!uunet!lll-winken!lll-tis!helios.ee.lbl.gov!pasteur!agate!
saturn!ssyx!koreth

From: koreth@ssyx.ucsc.edu (Steven Grimm)

Newsgroups: comp.sources.atari.st
Subject: v01i054: errorchk -- verbose display of error codes
Keywords: GST, 68000
Message-ID: <3763@saturn.ucsc.edu>
Date: 14 Jun 88 16:05:30 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 98
Approved: koreth@ssyx.ucsc.edu

Submitted-by: uunet!mcvax!philmds!leo (Leo de Wit)
Posting-number: Volume 1, Issue 54

Archive-name: errorchk

Here is another small memory-resident program to be placed in the AUTO folder.
    Did you also had times you wanted to see a program's final (error?) message when it flashed over your screen and the desktop was painted all over it.
    This little program checks whenever a program exits if the caller was the desktop (it does this by following the links in the basepages); if so, a message is printed and you can press a key to continue.
    The program sets the GEM program exit vector that is at address 0x408; this vector usually points to a RTS statement (i.e. does nothing). After a keypress a jump is made to the old vector (in case it WAS used).
    This program also prevents the need of coding similar messages into every program that can be called from the desktop.
    It was assembled using the assembler and linker that come with the GST-C compiler, resulting in a program length of 240 bytes.

    My name and address:

   L.J.M. de Wit
   Nachtegaallaan 7
   5731XP Mierlo
   Holland.

---------- H E R E I T A L L S T A R T S ----------

```
      MODULE  DTOPWAIT
      SECTION S.CCODE

GEMDOS    EQU 1
BIOS      EQU 13
PTERMRES  EQU $31
SETEXC    EQU 5
ESC       EQU $1B

DTWINIT
      MOVE.L   #-1,-(SP)
      MOVE.W   #$102,-(SP)
      MOVE.W   #SETEXC,-(SP)
      TRAP     #BIOS             * Save old GEM exit vector
      ADDQ.L   #8,SP
      LEA.L    OLDVEC(PC),A0
      MOVE.L   D0,(A0)
      PEA      DTOPWAIT
      MOVE.W   #$102,-(SP)
      MOVE.W   #SETEXC,-(SP)
      TRAP     #BIOS             * Set new GEM exit vector
      ADDQ.L   #8,SP
      MOVE.L   4(SP),A0
      MOVE.L   #$100,D0          * Base page
      ADD.L    12(A0),D0         * + text length
      ADD.L    20(A0),D0         * + data length
      ADD.L    28(A0),D0         * + bss length
```

```
      CLR.W    -(SP)             * Return value: 0 for success
      MOVE.L   D0,-(SP)          * # bytes to keep
      MOVE.W   #PTERMRES,-(SP)   * Keep process
      TRAP     #GEMDOS           * Stops here...

DTOPWAIT
      MOVE.L   $602C,A0
      MOVEQ.L  #0,D1
      BRA.S    DTOPAREND
DTOPARNT
      ADDQ.L   #1,D1
      MOVE.L   36(A0),A0
DTOPAREND
      TST.L    (A0)
      BNE.S    DTOPARNT
      CMP.B    #3,D1
      BNE.S    DTWEND
      LEA      WAITMSG(PC),A4
      BRA.S    DTOPW3
DTOPW2
      EXT.W    D0
      MOVE.W   D0,-(SP)
      MOVE.W   #2,-(SP)
      MOVE.W   #3,-(SP)
      TRAP     #BIOS
      ADDQ.L   #6,SP
DTOPW3
      MOVE.B   (A4)+,D0
      BNE.S    DTOPW2
      MOVE.W   #2,-(SP)
      MOVE.W   #2,-(SP)
      TRAP     #BIOS
      ADDQ.L   #4,SP
DTWEND
      MOVE.L   OLDVEC(PC),-(SP)
      RTS

      SECTION  S.DATA

OLDVEC   DC.L  0
WAITMSG  DC.B  13,10,ESC,'K',ESC,'p',9,9
         DC.B  'Program finished; hit a key to return to desktop',ESC,'q',0

      END
```

```
From 7723310559065907436
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,454e4c32b9c804e7,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-06-14 09:07:08 PST
Path: utzoo!attcan!uunet!husc6!think!ames!pasteur!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
```

Newsgroups: comp.sources.atari.st
Subject: v01i055: joinprnt -- echo console output to printer
Keywords: GST, 68000
Message-ID: <3769@saturn.ucsc.edu>
Date: 14 Jun 88 16:07:08 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 128
Approved: koreth@ssyx.ucsc.edu

Submitted-by: uunet!mcvax!philmds!leo (Leo de Wit)
Posting-number: Volume 1, Issue 55
Archive-name: joinprnt

The following nice little program (JOINPRNT.PRG) should be placed in the AUTO folder, so that it is made resident at startup time.
After pressing ^P (control-P) all screen output (but not the graphic part) is also directed to the printer. In this way every program can have its hardcopy without any trouble. Pressing ^P again toggles back to normal behaviour.
A few remarks:
A) The BIOS vector is altered to achieve this. This is however done transparently (using the old vector when necessary).
B) VT52 escape sequences are not send to the printer (the ESC and the following character are ignored).
C) The ^P has to be read first (by whatever program; presumably a shell); this is generally not a problem.
D) The program has been used by myself for about a month now, without problems.

The program was assembled and linked with the assembler and linker that come with the GST-C compiler, using the commands:
ASM.PRG JOINPRNT
LINK.PRG JOINPRNT
The resulting code was 246 bytes in size.

L.J.M. de Wit
Nachtegaallaan 7
5731XP Mierlo
Holland

---------- H E R E I T A L L S T A R T S ----------

```
      MODULE  JOINPR
      SECTION S.CCODE

GEMDOS    EQU 1
BIOS      EQU 13
PTERMRES  EQU $31
SETEXC    EQU 5
PTERM     EQU $4c
JOINON    EQU 0
HADESC    EQU 1
```

```
JOINIT
      MOVE.L   #-1,-(SP)
      MOVE.W   #$2D,-(SP)
      MOVE.W   #SETEXC,-(SP)
      TRAP     #BIOS             * Save old BIOS vector
      ADDQ.L   #8,SP
      LEA.L    OLDVEC(PC),A0
      MOVE.L   D0,(A0)
      PEA      JOINPR
      MOVE.W   #$2D,-(SP)
      MOVE.W   #SETEXC,-(SP)
      TRAP     #BIOS             * Set new BIOS vector
      ADDQ.L   #8,SP
      MOVE.L   4(SP),A0
      MOVE.L   #$100,D0          * Base page
      ADD.L    12(A0),D0         * + text length
      ADD.L    20(A0),D0         * + data length
      ADD.L    28(A0),D0         * + bss length
      CLR.W    -(SP)             * Return value: 0 for success
      MOVE.L   D0,-(SP)          * # bytes to keep
      MOVE.W   #PTERMRES,-(SP)   * Keep process
      TRAP     #GEMDOS           * Stops here...

* Entry point for 'embedded TRAP #13' in JOINPR
OLDTRAP
      MOVE.W   SR,-(SP)
* Entry point if not
OLDGEM
      MOVE.L   OLDVEC(PC),-(SP)
      RTS

* Altered BIOS routine
JOINPR
      LEA.L   FLAGS(PC),A1       * A1 will point to flags block
      MOVE.L  USP,A0             * A0 points to parameters to TRAP #13
      BTST    #5,(A7)
      BEQ.S   JOINUSER
      LEA     6(A7),A0           * If called from supervisor mode
JOINUSER
      CMP.W    #2,2(A0)          * console ?
      BNE.S    OLDGEM
      CMPI.W   #2,(A0)           * Bconin?
      BEQ.S    BCONIN
      CMPI.W   #3,(A0)           * Not Bconout
      BNE.S    OLDGEM
      TST.B    JOINON(A1)        * Flag set?
      BEQ.S    OLDGEM
      TST.B    HADESC(A1)        * Escape set?
      SF       HADESC(A1)        * reset flag first
      BNE.S    OLDGEM            * If flag was set
      CMP.W    #$1B,4(A0)        * Escape char?
      SEQ      HADESC(A1)
      BEQ.S    OLDGEM            * If escape
```

```
      MOVE.W   4(A0),-(SP)       * Make an extra copy of the word to print
      MOVE.W   4(A0),-(SP)
      MOVE.W   #0,-(SP)
      MOVE.W   #3,-(SP)
      BSR.S    OLDTRAP
      ADDQ.L   #6,SP
      MOVE.W   #2,-(SP)
      MOVE.W   #3,-(SP)
      BSR.S    OLDTRAP
      ADDQ.L   #6,SP
      RTE

BCONIN
      MOVE.W   #2,-(SP)
      MOVE.W   #2,-(SP)
      BSR.S    OLDTRAP
      ADDQ.L   #4,SP
      CMP.W    #'P'-'@',D0       * Control P?
      BNE.S    BCONEND
      LEA.L    FLAGS(PC),A1      * A1 points again to flags block
      NOT.B    JOINON(A1)        * Toggle flag
      BRA      JOINPR            * This discards current char
BCONEND
      RTE

      SECTION  S.DATA

OLDVEC   DC.L  0
FLAGS    DC.W  0                 * Two flag bytes: JOINON and HADESC

      END
```

```
From 6778866786036987461
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,fd5567673a84833e,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-06-14 10:02:23 PST
Path: utzoo!attcan!uunet!husc6!uwvax!vanvleck!uwmcsd1!ig!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i056: color -- allow user to edit color palette
Keywords: Laser C, low, medium
Message-ID: <3773@saturn.ucsc.edu>
Date: 14 Jun 88 17:02:23 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 538
Approved: koreth@ssyx.ucsc.edu

Submitted-by: uunet!van-bc!resrch!bobs (Robert J. Silliker)
Posting-number: Volume 1, Issue 56
Archive-name: color
```

[This is the source to a dialog box handler that lets the user
 modify the color palette. Resource file is included and will
 be uudecoded automatically by the shar file. -sg]

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      COLOR.C
#      COLOR.DEF
#      COLOR.DOC
#      COLOR.H
#      COLOR.RSC
#      MAKEFILE
#      README
#      TEST.C
#
sed 's/^X//' << 'SHAR_EOF' > COLOR.C &&
X/*************************************************/
X/*                                   */
X/*                COLOR.C            */
X/*                                   */
X/*  Copyright (c) 1988, by Robert Joseph Silliker  */
X/*              All Rights Reserved          */
X/*      Permission is granted for unrestricted     */
X/*            non-commercial use             */
X/*                                   */
X/*************************************************/
X
X#include <osbind.h>
X#include <gemdefs.h>
X#include <obdefs.h>
X#include "color.h"
X
X#include "objects.c"
X
X
Xint work_rgb[16][3];  /* Current settings for the colors */
Xint old_rgb[16][3];   /* Color settings at initialization */
Xint color_set = 0;
Xint color_arrow[] = {ARROW0, ARROW1, ARROW2, ARROW3, ARROW4, ARROW5,
X             ARROW6, ARROW7, ARROW8, ARROW9, ARROW10, ARROW11,
X             ARROW12, ARROW13, ARROW14, ARROW15};
X
Xint color_buttons[3][8] = {{COLOR1_8, COLOR1_7, COLOR1_6, COLOR1_5,
X                 COLOR1_4, COLOR1_3, COLOR1_2, COLOR1_1},
X                {COLOR2_8, COLOR2_7, COLOR2_6, COLOR2_5,
X                 COLOR2_4, COLOR2_3, COLOR2_2, COLOR2_1},
X                {COLOR3_8, COLOR3_7, COLOR3_6, COLOR3_5,
X                 COLOR3_4, COLOR3_3, COLOR3_2, COLOR3_1}};
X
Xint color_levels[] = {0, 213, 356, 499, 642, 785, 928, 1000};
Xint mono_levels[] = {0, 1000};
```

```
X
X#define HIRES 2
X#define MEDRES 1
X#define LORES 0
X
Xstatic int set_columns();
Xstatic int set_col();
Xstatic int copy_color();
X
Xstatic int max_colors;
Xstatic int init_done = 0;
Xstatic int mode;
Xstatic OBJECT *color_object;
X
X/*-----------------------------------*/
X/* Display the color dialog and allow */
X/* the user to interact with it.      */
X/*-----------------------------------*/
X
Xdo_color(handle)
Xint handle;
X{
X  int save_rgb[16][3];
X  register int i, j, ob;
X  int oldob;
X
X  if(!init_done) {
X    mode = Getrez();
X
X    if(mode == LORES)
X      max_colors = 16;
X    else if(mode == MEDRES)
X      max_colors = 4;
X    else
X      max_colors = 1;
X
X    /*-------------------------------------------------------*/
X    /* Convert the objects from char coords to pixel coords */
X    /*-------------------------------------------------------*/
X
X    b_fix_objects();
X
X    color_object = b_rs_trindex[TSETCOLR];
X  }
X
X  copy_color(save_rgb, work_rgb);
X  set_columns(color_set, 0);
X
X  chg_obj(color_object, color_arrow[0], NORMAL, 0);
X
X  for(i=1; i<4; i++) {
X    if(mode == HIRES)
X      chg_obj(color_object, color_arrow[i], DISABLED, 0);
```

```
X    else
X      chg_obj(color_object, color_arrow[i], NORMAL, 0);
X  }
X
X  for(i=4; i<16; i++) {
X    if(mode != LORES)
X      chg_obj(color_object, color_arrow[i], DISABLED, 0);
X    else
X      chg_obj(color_object, color_arrow[i], NORMAL, 0);
X  }
X
X  oldob = color_arrow[color_set];
X  chg_obj(color_object, oldob, OUTLINED, 0);
X
X  if(!init_done) {
X    form_center(color_object, &color_object->ob_x, &color_object->ob_y,
X                            &color_object->ob_width, &color_object->ob_height);
X  }
X
X  form_dial(FMD_START, 0, 0, 0, 0, color_object->ob_x-3,
X                    color_object->ob_y-3,
X                    color_object->ob_width+6,
X                    color_object->ob_height+6);
X
X  objc_draw(color_object, 0, MAX_DEPTH, color_object->ob_x-3,
X                      color_object->ob_y-3,
X                      color_object->ob_width+6,
X                      color_object->ob_height+6);
X
X  do {
X    ob = form_do(color_object, 0);
X    if(ob == COLORCAN) {
X      for(i=0; i<3; i++)
X        set_col(i, color_set, NORMAL, 0);
X      copy_color(work_rgb, save_rgb);
X
X      for(i=0; i<max_colors; i++)
X        vs_color(handle, i, &work_rgb[i][0]);
X    }
X    else if(ob != COLOROK) {
X      for(i=0; i<max_colors; i++) {
X        if(color_arrow[i] == ob) {
X          set_columns(i, 1);
X          break;
X        }
X      }
X      if(i < max_colors) {
X        (color_object+oldob)->ob_state = NORMAL;
X        (color_object+ob)->ob_state = OUTLINED;
X
X    draw_button(ob);
X    draw_button(oldob);
X
```

```
X        oldob = ob;
X      }
X      else {
X        for(i=0; i<3; i++) {
X          for(j=0; j<8; j++) {
X            if(color_object[color_buttons[i][j]].ob_state & SELECTED) {
X              work_rgb[color_set][i] = color_levels[j];
X              break;
X            }
X          }
X        }
X        vs_color(handle, color_set, &work_rgb[color_set][0]);
X      }
X    }
X  } while((ob != COLOROK) && (ob != COLORCAN));
X
X  chg_obj(color_object, ob, NORMAL, 0);
X
X  form_dial(FMD_FINISH, 0, 0, 0, 0, color_object->ob_x-3,
X                       color_object->ob_y-3,
X                       color_object->ob_width+6,
X                       color_object->ob_height+6);
X
X  init_done = 1;
X}
X
X/*-----------------------------*/
X/* Save or restore the colors. */
X/* Flag  = 0 - Restore.        */
X/* Flag != 0 - Save.           */
X/*-----------------------------*/
X
Xsave_color(handle, flag)
Xint handle, flag;
X{
X  register int i;
X
X  for(i=0; i<16; i++) {
X    if(flag) {
X      vq_color(handle, i, 0, &work_rgb[i][0]);
X      vq_color(handle, i, 0, &old_rgb[i][0]);
X    }
X    else
X      vs_color(handle, i, &old_rgb[i][0]);
X  }
X}
X
Xstatic int set_columns(color, flag)
Xint color, flag;
X{
X  register int i;
X
X  for(i=0; i<3; i++)
```

```
X    set_col(i, color_set, NORMAL, flag);
X
X  for(i=0; i<3; i++)
X    set_col(i, color, SELECTED, flag);
X
X  color_set = color;
X}
X
Xstatic int set_col(index, color, state, flag)
Xint index, color;
Xint state, flag;
X{
X  register int color_index;
X
X  color_index = (work_rgb[color][index] + 72) / 143;
X
X  if(color_index < 0)
X    color_index = 0;
X  else if(color_index > 7)
X    color_index = 7;
X
X  chg_obj(color_object, color_buttons[index][color_index], state, flag);
X}
X
Xstatic int copy_color(d, s)
Xint d[][3], s[][3];
X{
X  register int i, j;
X
X  for(i=0; i<16; i++) {
X    for(j=0; j<3; j++)
X      d[i][j] = s[i][j];
X  }
X}
X
X/*
Xcenter_form(p)
XOBJECT *p;
X{
X  form_center(p, &p->ob_x, &p->ob_y, &p->ob_width, &p->ob_height);
X}
X
Xdraw_obj(p, ob)
Xregister OBJECT *p;
Xint ob;
X{
X  objc_draw(p, ob, MAX_DEPTH, p->ob_x, p->ob_y,
X                                      p->ob_width, p->ob_height);
X}
X
Xdial_obj(flag, p)
Xint flag;
XOBJECT *p;
```

```
X{
X  form_dial(flag, 0, 0, 0, 0, p->ob_x, p->ob_y,
X                                       p->ob_width, p->ob_height);
X}
X*/
X
Xchg_obj(p, ob, state, flag)
XOBJECT *p;
Xint ob, state, flag;
X{
X  objc_change(p, ob, 0, p->ob_x, p->ob_y,
X                                  p->ob_width, p->ob_height, state, flag);
X}
X
Xdraw_button(ob)
Xint ob;
X{
X  int x, y;
X
X  objc_offset(color_object, ob, &x, &y);
X
X  objc_draw(color_object, ARROWBOX, MAX_DEPTH, x-3, y-3,
X            (color_object+ob)->ob_width+6, (color_object+ob)->ob_height+6);
X}
X
X/* END OF FILE */
SHAR_EOF
chmod 0600 COLOR.C || echo "restore of COLOR.C fails"
sed 's/^X//' << 'SHAR_EOF' > COLOR.DEF &&
XSHAR_EOF
chmod 0600 COLOR.DEF || echo "restore of COLOR.DEF fails"
sed 's/^X//' << 'SHAR_EOF' > COLOR.DOC &&
XCOLOR (3L)                                COLOR(3L)
X
X
XNAME
X    COLOR - Library routines to manipulate the color palette from
X         within a 'C' program.
X
XAUTHOR
X    Bob Silliker
X
XCOPYRIGHT
X    Copyright (c) 1988 by Robert Joseph Silliker
X    All Rights Reserved
X    Permission is granted for unrestricted non-commercial use
X
XSYNOPSIS
X    save_color(handle, flag)
X    int handle, flag;
X
X    do_color(handle)
X    int handle;
```

```
X
XDESCRIPTION
X    These library routines provide a means of manipulating the color
X        palette through a dialog box.
X
X    Save_color() is used to  save/restore the  current color pallett
X    settings into/from a global rgb array called old_rgb[][3].  This
X    usually is called at the start of the application with a flag
X    value of 1.  This causes the current settings to be saved.  When
X        the program terminates, this routine should be called with a flag
X        value of 0.  This will cause the current  settings to be restored.
X        Handle is what you get from opening a virtual workstation.
X
X    Do_color() displays a dialog and allows the user to set the RGB
X    of any of the active colors.  The user can cancel the operation
X    which will restore the colors to what they were before do_color()
X    was called.  The default exit button is CANCEL so if the user
X    manages to set foreground and background colors the same pressing
X    the <RETURN> key will cancel the operation.
X
XWARNING
X    Save_color() must be called  with a flag value of 1 before a call
X    is made to do_color().  If you don't then all colors will be set
X    to 0.
SHAR_EOF
chmod 0600 COLOR.DOC || echo "restore of COLOR.DOC fails"
sed 's/^X//' << 'SHAR_EOF' > COLOR.H &&
X#define TSETCOLR 1
X#define COLOR1_2 3
X#define COLOR1_1 2
X#define COLOR1_3 4
X#define COLOR1_4 5
X#define COLOR1_6 7
X#define COLOR1_5 6
X#define COLOR1_7 8
X#define COLOR1_8 9
X#define COLOR2_1 11
X#define COLOR2_7 12
X#define COLOR2_6 13
X#define COLOR2_3 15
X#define COLOR2_4 14
X#define COLOR2_2 16
X#define COLOR2_5 17
X#define COLOR2_8 18
X#define COLOR3_1 20
X#define COLOR3_8 21
X#define COLOR3_6 23
X#define COLOR3_7 22
X#define COLOR3_5 24
X#define COLOR3_4 25
X#define COLOR3_2 27
X#define COLOR3_3 26
X#define ARROWBOX 28
```

```
X#define ARROW0 29
X#define ARROW4 31
X#define ARROW2 30
X#define ARROW6 32
X#define ARROW8 33
X#define ARROW12 35
X#define ARROW10 34
X#define ARROW14 36
X#define ARROW1 37
X#define ARROW5 39
X#define ARROW3 38
X#define ARROW7 40
X#define ARROW9 41
X#define ARROW13 43
X#define ARROW11 42
X#define ARROW15 44
X#define COLOROK 48
X#define COLORCAN 49
SHAR_EOF
chmod 0600 COLOR.H || echo "restore of COLOR.H fails"
sed 's/^X//' << 'SHAR_EOF' > s2_temp_.tmp &&
Xbegin 600 COLOR.RSC
XM````+@`N`"X`+@`N`"0`+@`N!-X`,@`"``````````````3F3TL`0T%.0T5,
XM`/__``$`,0`4```$``"$0``````!L`$0`*``(`"0`4`````#_$0``!``!
XM``,`"``#_____P`4`!4```!$0````````,``0`$_____P`4`!4````!$0``
XM```!``,``0`%_____P`4`!4````!$0`````"``,``0`&_____P`4`!4````!
XM$0`````#``,``0`'_____P`4`!4```!$0````$``,``0`(_____P`4`!4`
XM```!$0````%``,``0`)_____P`4`!4```!$0````&``,``0`!_____P`4
XM`!4````!$0`````'``,``0`3``L`$@`4`````#_$0``#``!``,`"``,____
XM_P`4`!4```!$0````````,``0`-_____P`4`!4```!$0````&``,``0`.
XM_____P`4`!4````!$0````%``,``0`/_____P`4`!4````!$0````#``,`
XM`0`0_____P`4`!4````!$0`````"``,``0`1_____P`4`!4```!$0`````!
XM``,``0`2_____P`4`!4```!$0````$``,``0`*_____P`4`!4````!$0``
XM```'``,``0`<`!0`&P`4`````#_$0``%``!``,`"``5_____P`4`!4````!
XM$0````````,``0`6_____P`4`!4````!$0`````'``,``0`7_____P`4`!4`
XM```!$0````&``,``0`8_____P`4`!4```!$0````%``,``0`9_____P`4
XM`!4````!$0````$``,``0`:_____P`4`!4```!$0````#``,``0`;____
XM_P`4`!4```!$0`````"``,``0`3_____P`4`!4```!$0````!``,``0`M
XM`!T`+``4````````$4```0`*`!D`!0`>_____P`4``4```#_$0```0`!``(`
XM`0`?_____P`4``4```#_$7(`!``!``(``0`@_____P`4``4```#_$70`!P`!
XM``(``0`A_____P`4``4```#_$78"@`!``(``0`B_____P`4``4```#_$7@`
XM#0`!``(``0`C_____P`4``4```#_$7H`$``!``(``0`D_____P`4``4```#_
XM$7P`$P`!``(``0`E_____P`4``4```#_$7X`%@`!``(``0`F_____P`4``4`
XM``#_$7$``0`#``(``0`G_____P`4``4```#_$7,`!``#``(``0`H_____P`4
XM``4```#_$74`!P`#``(``0`I_____P`4``4```#_$7<`"@`#``(``0`J____
XM_P`4``4```#_$7D`#0`#``(``0`K_____P`4``4```#_$7L`$``#``(``0`L
XM_____P`4``4```#_$7T`$P`#``(``0`<_____P`4``4```#_$7\`%@`#``(`
XM`0`N_____P`;````%+_$0``!``)``,``0`O_____P`;````$?_$0``#``)
XM``,``0`P_____P`;````$+_$0``%``)``,``0`Q_____P`:``4``````"0`
XG`@`/``@``0``_____P`:`"<``````"<`$0`/``@``0```"X```N
X`
Xend
SHAR_EOF
```

```
uudecode < s2_temp_.tmp && rm -f s2_temp_.tmp &&
chmod 0600 COLOR.RSC || echo "restore of COLOR.RSC fails"
sed 's/^X//' << 'SHAR_EOF' > MAKEFILE &&
X#-------------------------------------------------------------------------
X#         Makefile to make the library routine 'color.o'
X#-------------------------------------------------------------------------
X
XOBJECT = color.prg
XFILES  = color.o test.o
X
Xcolor.o: color.c objects.c
X
X$(FILES): color.h
X
Xobjects.c: color.rsc
X    \bin\rsctoc.ttp color.rsc -p b_ -o objects.c
X
X$(OBJECT): $(FILES)
X    $(CC) $(FILES) -o $(OBJECT)
X
Xclean:
X    $(RM) $(FILES)
X    $(RM) $(OBJECT)
X
X#-------------------------------------------------------------------------
SHAR_EOF
chmod 0600 MAKEFILE || echo "restore of MAKEFILE fails"
sed 's/^X//' << 'SHAR_EOF' > README &&
XThis is the source for a dialog to play with the color palette.  It was
Xintended that this be an example of how to the the 'rsctoc' program.
XHowever, as distributed the source can be used without 'rsctoc'.  This
Xwas compiled under Laser C and will probably need some mods to work
Xwith another compiler.
X
Xcolor.doc - manual page for the color library routines.
X
Xtest.c    - is a test program that calls the routines provided by color.
X
Xcolor.c   - is the source code that handles the dialog box.
X
Xobjects.c - is the object tree for the color dialog and was generated
X           by running the 'rsctoc' program on the 'color.rsc' file.
X           See the makefile for command line args.
X
Xcolor.rsc - resource file for the color dialog.
X
Xcolor.h   - header file created by the Megamax RCP program.
X
Xcolor.def - another file created by the Megamax RCP program that contains
X           information about the object names and types.
X
Xmakefile  - makefile that I use with my version of make.  This will probably
X           need some changes.
```

```
X
X
XBob Silliker
SHAR_EOF
chmod 0600 README || echo "restore of README fails"
sed 's/^X//' << 'SHAR_EOF' > TEST.C &&
X/*************************************************/
X/*                                   */
X/*               COLOR.C             */
X/*                                   */
X/*  Copyright (c) 1988, by Robert Joseph Silliker  */
X/*             All Rights Reserved           */
X/*      Permission is granted for unrestricted     */
X/*            non-commercial use             */
X/*                                   */
X/*************************************************/
X
X#include <osbind.h>
X#include <gemdefs.h>
X#include <obdefs.h>
X#include "color.h"
X
Xint intin[128];
Xint intout[128];
Xint ptsin[128];
Xint ptsout[128];
Xint contrl[12];
X
Xmain()
X{
X  int i, handle;
X  int work_in[11],work_out[57];
X
X  appl_init();
X
X  handle = graf_handle(&i, &i, &i, &i);
X
X  for (i = 0; i < 10; ++i)
X    work_in[i] = 1;
X  work_in[10] = 2;
X
X  v_opnvwk(work_in, &handle, work_out);
X
X  graf_mouse(ARROW, 0);
X
X  save_color(handle, 1);
X  do_color(handle);
X  save_color(handle, 0);
X
X  v_clsvwk(handle);
X  appl_exit();
X}
X
```

```
X/* END OF FILE */
SHAR_EOF
chmod 0600 TEST.C || echo "restore of TEST.C fails"
exit 0


From -2685331090641726343
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,65cd7d864cd5a84a,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-06-14 10:02:58 PST
Path: utzoo!attcan!uunet!husc6!bbn!uwmcsd1!ig!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i057: rsctoc -- convert .RSC file to .C file
Keywords: Laser C
Message-ID: <3774@saturn.ucsc.edu>
Date: 14 Jun 88 17:02:58 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 799
Approved: koreth@ssyx.ucsc.edu

Submitted-by: uunet!van-bc!resrch!bobs (Robert J. Silliker)
Posting-number: Volume 1, Issue 57
Archive-name: rsctoc

[This takes a resource file and generates the equivalent
 C source file. -sg]

#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      MAIN.C
#      MAKEFILE
#      README
#      RSCTOC.DOC
#
sed 's/^X//' << 'SHAR_EOF' > MAIN.C &&
X/*************************************************/
X/*                                   */
X/*               RSCTOC  V1.0         */
X/*                                   */
X/*  Copyright (c) 1988, by Robert Joseph Silliker  */
X/*             All Rights Reserved          */
X/*      Permission is granted for unrestricted     */
X/*            non-commercial use            */
X/*                                   */
X/*************************************************/
X
X#include <stdio.h>
X#include <fcntl.h>
X#include <osbind.h>
```

```
X#include <gemdefs.h>
X#include <obdefs.h>
X
Xstatic char *copyright[] = "RSCTOC V1.0\nCopyright (c) 1988, by\
X Robert Joseph Silliker\nAll Rights Reserved\n\
X Permission is granted for unrestricted non-commercial use\n";
X
Xlong _stksize = 16000;
X
XRSHDR *rschead;            /* Pointer to the resource file header. */
Xchar *out_path = NULL;     /* Path to output file. */
Xchar *rsc_path = NULL;     /* Path to input file.  */
XFILE *fp = stdout;         /* Default output file. */
Xchar *prefix= "";     /* Prefix to the names of arrays. */
X
X#define MAXTREES 512       /* Maximum number of object trees per file. */
X
XOBJECT *objects;      /* Pointer to the base of the objects. */
XOBJECT *trindex[MAXTREES]; /* One pointer per object tree. */
X
Xchar *progname = "rsctoc.ttp";  /* Program name (no argv[0]) */
X
X
Xmain(argc, argv)
Xint argc;
Xchar *argv[];
X{
X  int i;
X  extern char *rsc_path, *out_path, *prefix;
X  extern FILE *fp;
X
X  if((argc != 2) && (argc != 4) && (argc != 6))
X    usage("Incorrect number of arguments.");
X
X  rsc_path = argv[1];
X
X  for(i=2; i<argc; i+=2) {     /* This lets option arguments like */
X    if(!strcmp(argv[i], "-p"))      /* rsctoc.ttp rsc_file -f file1 -f file2 */
X      prefix = argv[i+1];      /* rsctoc.ttp rsc_file -p XXXX -p YYYY */
X    else if(!strcmp(argv[i], "-o")) /* without a usage error. */
X      out_path = argv[i+1];
X    else
X      usage("Illegal option.");
X  }
X
X  if(!init_prog()) {
X    put_header();
X    make_tedinfo();
X    make_icon();
X    make_bitblk();
X    make_object();
X    make_index();
X    make_fix();
```

```
X  }
X
X  clean_up();
X}
X
X
X/*---------------------------------------*/
X/* Output the usage message and terminate */
X/* the program.                          */
X/*---------------------------------------*/
X
Xusage(s)
Xchar *s;
X{
X  extern char *progname;
X
X  if(s)
X    fprintf(stderr, "%s: %s\n", progname, s);
X
X  fprintf(stderr, "Usage: %s rsc_file [-p prefix] [-o outfile]\n", progname);
X
X  exit(2);
X}
X
X
X/*-----------------------------------------*/
X/* Output an error message and terminate the */
X/* program gracefully if the caller sets the */
X/* terminate flag.  The value of the flag    */
X/* used to terminate is returned by the      */
X/* program.                                  */
X/*-----------------------------------------*/
X
Xmperror(terminate, s)
Xint terminate;
Xchar *s;
X{
X  extern char *progname;
X
X  fprintf(stderr, "%s: %s\n", progname, s);
X
X  if(terminate) {
X    clean_up();
X    exit(terminate);
X  }
X}
X
X
X/*-------------------------*/
X/* Initialize the program. */
X/*-------------------------*/
X
Xint init_prog()
```

```
X{
X  int i;
X  extern char *rsc_path, *out_path;
X  extern FILE *fp;
X  extern RSHDR *rschead;
X
X  if(!myrsrc_load(rsc_path))
X    mperror(1, "Error getting resource file.");
X
X  if(out_path != NULL) {
X    if((fp = fopen(out_path, "w")) == NULL)
X      mperror(1, "Can't open output file.");
X  }
X
X  if(rschead->rsh_nobs <= 0)
X    mperror(1, "No object trees to dump.");
X
X  return(0);
X}
X
X
X/*-------------------------------------------*/
X/* Clean up open files, close the workstation */
X/* and close the aes application.             */
X/*-------------------------------------------*/
X
Xclean_up()
X{
X  extern FILE *fp;
X
X  if((fp != stdout) && (fp != NULL))
X    fclose(fp);
X}
X
X
X/*----------------------------*/
X/* Extract tedinfo information. */
X/*----------------------------*/
X
Xmake_tedinfo()
X{
X  int i, first_one;
X  register OBJECT *g;
X  TEDINFO *t;
X  extern char *prefix;
X  extern FILE *fp;
X  extern OBJECT *objects;
X  extern RSHDR *rschead;
X
X  fprintf(fp, "TEDINFO %srs_tedinfo[] = {\n", prefix);
X
X  for(i=0, first_one=1, g=objects; i < rschead->rsh_nobs; i++, g++) {
X    if(g->ob_type == G_FTEXT || g->ob_type == G_FBOXTEXT ||
```

```
X       g->ob_type == G_TEXT  || g->ob_type == G_BOXTEXT) {
X
X      if(first_one) {
X    first_one = 0;
X      }
X      else
X        fprintf(fp, ",\n");
X
X      t = (TEDINFO *)(g->ob_spec);
X
X      if(strlen(t->te_ptext))
X        fprintf(fp, "  {\"%s\", ", t->te_ptext);
X      else
X        fprintf(fp, "  {%sp_to_nul, ", prefix);
X
X      if(strlen(t->te_ptmplt))
X        fprintf(fp, "\"%s\", ", t->te_ptmplt);
X      else
X        fprintf(fp, "%sp_to_nul, ", prefix);
X
X      if(strlen(t->te_pvalid))
X        fprintf(fp, "\"%s\",\n  ", t->te_pvalid);
X      else
X        fprintf(fp, "%sp_to_nul,\n   ", prefix);
X
X      fprintf(fp, "%3d, ", t->te_font);
X      fprintf(fp, "%3d, ", t->te_junk1);
X      fprintf(fp, "%3d, ", t->te_just);
X      fprintf(fp, "%3d, ", t->te_color);
X      fprintf(fp, "%3d, ", t->te_junk2);
X      fprintf(fp, "%3d, ", t->te_thickness);
X      fprintf(fp, "%3d, ", t->te_txtlen);
X      fprintf(fp, "%3d}", t->te_tmplen);
X    }
X  }
X
X  if(!first_one)
X    fprintf(fp, "\n};\n\n");
X  else
X    fprintf(fp, "  0L\n};\n\n");
X}
X
X
X/*---------------------------*/
X/* Extract icon information. */
X/*---------------------------*/
X
Xmake_icon()
X{
X  int i, j, k, numb;
X  int first_one, *p;
X  register OBJECT *g;
X  extern char *prefix;
```

```
X  ICONBLK *icon;
X  extern FILE *fp;
X  extern OBJECT *objects;
X  extern RSHDR *rschead;
X
X  for(i=0, numb=0, g=objects; i < rschead->rsh_nobs; i++, g++) {
X    /*----------------------*/
X    /* First do the images. */
X    /*----------------------*/
X
X    if(g->ob_type == G_ICON) {
X      icon = (ICONBLK *)(g->ob_spec);
X
X      fprintf(fp, "int %simask%03d[] = {", prefix, numb);
X      k = icon->ib_hicon * (icon->ib_wicon/16);
X
X      for(j=0, p=icon->ib_pmask; k > 0; j++, p++, k--) {
X        if(!(j % 8))
X          fprintf(fp, "\n  ");
X        fprintf(fp, "0x%04x", *p);
X
X    if(k != 1)
X          fprintf(fp, ",");
X      }
X      fprintf(fp, "\n};\n\n");
X
X      fprintf(fp, "int %sidata%03d[] = {", prefix, numb);
X      k = icon->ib_hicon * (icon->ib_wicon/16);
X
X      for(j=0, p=icon->ib_pdata; k > 0; j++, p++, k--) {
X        if(!(j % 8))
X      fprintf(fp, "\n   ");
X        fprintf(fp, "0x%04x", *p);
X
X        if(k != 1)
X      fprintf(fp, ",");
X      }
X
X      fprintf(fp, "\n};\n\n");
X      numb++;
X    }
X  }
X
X  fprintf(fp, "ICONBLK %srs_iconblk[] = {\n", prefix);
X
X  for(i=0, numb=0, g=objects, first_one=1; i < rschead->rsh_nobs; i++, g++) {
X    /*-----------------------------*/
X    /* Now do the icon structures. */
X    /*-----------------------------*/
X
X    if(g->ob_type == G_ICON) {
X
X      if(first_one)
```

```
X       first_one = 0;
X     else
X       fprintf(fp, ",\n");
X
X     icon = (ICONBLK *)(g->ob_spec);
X
X     fprintf(fp, "  {%simask%03d, %sidata%03d, ", prefix, numb, prefix, numb);
X
X     if(strlen(icon->ib_ptext))
X       fprintf(fp, "\"%s\",\n", icon->ib_ptext);
X     else
X       fprintf(fp, "%sp_to_nul, \n", prefix);
X
X     fprintf(fp, "   0x%04x, 0x%04x, 0x%04x, 0x%04x, 0x%04x, 0x%04x,\n",
X             icon->ib_char, icon->ib_xchar, icon->ib_ychar,
X                 icon->ib_xicon, icon->ib_yicon, icon->ib_wicon);
X
X     fprintf(fp, "   0x%04x, 0x%04x, 0x%04x, 0x%04x, 0x%04x},",
X         icon->ib_hicon, icon->ib_xtext, icon->ib_ytext,
X         icon->ib_wtext, icon->ib_htext);
X     numb++;
X   }
X }
X
X if(!first_one)
X   fprintf(fp, "\n};\n\n");
X else
X   fprintf(fp, "  0L\n};\n\n");
X}
X
X
X
X/*-----------------------------*/
X/* Extract bitblk information. */
X/*-----------------------------*/
X
Xmake_bitblk()
X{
X int i, j, k, numb;
X int first_one, *p;
X register OBJECT *g;
X BITBLK *bitblk;
X extern char *prefix;
X extern FILE *fp;
X extern OBJECT *objects;
X extern RSHDR *rschead;
X
X for(i=0, numb=0, g=objects; i < rschead->rsh_nobs; i++, g++) {
X   /*----------------------*/
X   /* First do the images. */
X   /*----------------------*/
X
X   if(g->ob_type == G_IMAGE) {
```

```
X      bitblk = (BITBLK *)(g->ob_spec);
X
X      fprintf(fp, "int %sbblock%03d[] = {", prefix, numb);
X      k = bitblk->bi_hl * (bitblk->bi_wb/2);
X
X      for(j=0, p=bitblk->bi_pdata; k > 0; j++, p++, k--) {
X        if(!(j % 8))
X          fprintf(fp, "\n  ");
X        fprintf(fp, "0x%04x", *p);
X
X    if(k != 1)
X          fprintf(fp, ",");
X      }
X      fprintf(fp, "\n};\n\n");
X
X      numb++;
X    }
X  }
X
X  fprintf(fp, "BITBLK %srs_bitblk[] = {\n", prefix);
X
X  for(i=0, numb=0, g=objects, first_one=1; i < rschead->rsh_nobs; i++, g++) {
X    /*------------------------------*/
X    /* Now do the bitblk structures. */
X    /*------------------------------*/
X
X    if(g->ob_type == G_IMAGE) {
X
X      if(first_one)
X        first_one = 0;
X      else
X        fprintf(fp, ",\n");
X
X      bitblk = (BITBLK *)(g->ob_spec);
X
X      fprintf(fp, "  {%sbblock%03d, ", prefix, numb);
X
X      fprintf(fp, " 0x%04x, 0x%04x, 0x%04x, 0x%04x, 0x%04x}",
X                          bitblk->bi_wb, bitblk->bi_hl, bitblk->bi_x,
X                               bitblk->bi_y, bitblk->bi_color);
X
X      numb++;
X    }
X  }
X
X  if(!first_one)
X    fprintf(fp, "\n};\n\n");
X  else
X    fprintf(fp, "  0L\n};\n\n");
X}
X
X
X/*-----------------------------------------------*/
```

```
X/* Extract object information and change into C. */
X/*-------------------------------------------*/
X
Xmake_object()
X{
X  int i, ic, bb, ted;
X  int first_one;
X  register OBJECT *g;
X  extern char *prefix;
X  extern FILE *fp;
X  extern OBJECT *objects;
X  extern RSHDR *rschead;
X
X  for(i=0,ic=0,bb=0,ted=0,g=objects,first_one=1; i<rschead->rsh_nobs;i++,g++) {
X    if(first_one) {
X      fprintf(fp, "OBJECT %srs_object[] = {\n", prefix);
X      first_one = 0;
X    }
X    else
X      fprintf(fp, ",\n");
X
X    print_index_comment(g);
X
X    fprintf(fp, "  {%4d,%4d,%4d,0x%04x,0x%04x,0x%04x, ",
X                               g->ob_next, g->ob_head, g->ob_tail,
X                               g->ob_type, g->ob_flags, g->ob_state);
X
X    if(g->ob_type == G_STRING || g->ob_type == G_TITLE
X                              || g->ob_type == G_BUTTON) {
X
X      if(strlen(g->ob_spec))
X        fprintf(fp, "\"%s\",", g->ob_spec);
X      else
X        fprintf(fp, "%sp_to_nul,", prefix);
X    }
X    else if(g->ob_type == G_FTEXT || g->ob_type == G_FBOXTEXT
X                                  || g->ob_type == G_TEXT
X                                  || g->ob_type == G_BOXTEXT) {
X      fprintf(fp, "\&%srs_tedinfo[%d],", prefix, ted);
X      ted++;
X    }
X    else if(g->ob_type == G_ICON) {
X      fprintf(fp, "\&%srs_iconblk[%d],", prefix, ic);
X      ic++;
X    }
X    else if(g->ob_type == G_IMAGE) {
X      fprintf(fp, "\&%srs_bitblk[%d],", prefix, bb);
X      bb++;
X    }
X    else
X      fprintf(fp, "0x%08lxL,", g->ob_spec);
X
X    fprintf(fp, "%4d,%4d,%4d,%4d}",
```

```
X         g->ob_x, g->ob_y, g->ob_width, g->ob_height);
X  }
X
X  if(!first_one)
X    fprintf(fp, "\n};\n\n");
X}
X
X
X/*-----------------------------------------------*/
X/* Build the array of tree pointers and generate */
X/* the structure size macros.                    */
X/*-----------------------------------------------*/
X
Xmake_index()
X{
X  int i, index, first_one;
X  extern char *prefix;
X  extern FILE *fp;
X  extern OBJECT *objects, *trindex[];
X  extern RSHDR *rschead;
X
X  for(i=0, first_one=1; i < rschead->rsh_ntree; i++) {
X    if(first_one) {
X      fprintf(fp, "OBJECT *%srs_trindex[] = {\n", prefix);
X      first_one = 0;
X    }
X    else
X      fprintf(fp, ",\n");
X
X    index = (trindex[i] - objects);
X
X    fprintf(fp, "  &%srs_object[%d]", prefix, index);
X  }
X  if(!first_one)
X    fprintf(fp, "\n};\n\n");
X
X  fprintf(fp, "#define %sNUM_TI %d\n", prefix, rschead->rsh_nted);
X  fprintf(fp, "#define %sNUM_IB %d\n", prefix, rschead->rsh_nib);
X  fprintf(fp, "#define %sNUM_BB %d\n", prefix, rschead->rsh_nbb);
X  fprintf(fp, "#define %sNUM_OBS %d\n", prefix, rschead->rsh_nobs);
X  fprintf(fp, "#define %sNUM_TREE %d\n\n\n", prefix, rschead->rsh_ntree);
X}
X
X
X/*-----------------------------------------------------*/
X/* Output the include #ifdef for the object definition  */
X/* header file.  Output a single NUL character pointer  */
X/* into the C source.  This will be used as the pointer */
X/* to an empty character string wherever strings have   */
X/* zero length.                                         */
X/*-----------------------------------------------------*/
X
Xput_header()
```

```
X{
X  extern FILE *fp;
X
X  fprintf(fp, "#ifndef OBJECT\n");
X  fprintf(fp, "#include <obdefs.h>\n");
X  fprintf(fp, "#endif\n\n");
X  fprintf(fp, "static char %sp_to_nul[1] = \"\";\n\n", prefix);
X}
X
X
X/*----------------------------------------*/
X/* Output code that will fix up the objects. */
X/*----------------------------------------*/
X
Xmake_fix()
X{
X  extern char *prefix;
X  extern FILE *fp;
X
X  fprintf(fp, "%sfix_objects()\n{\n", prefix);
X  fprintf(fp, "  register int i;\n\n");
X  fprintf(fp, "  for(i=0; i<%sNUM_OBS; i++)\n", prefix);
X  fprintf(fp, "    rsrc_obfix(%srs_object, i);\n", prefix);
X  fprintf(fp, "}\n\n");
X}
X
X
X/*--------------------------------*/
X/* Open and read in the rsc file. */
X/* Then call the routine to fix   */
X/* the object pointers.           */
X/*--------------------------------*/
X
Xchar *pbase;          /* Pointer to base of file in memory */
X#define BLKSIZE 8192       /* Number of bytes per read call */
X                  /* Don't make this larger than 32767 */
X
Xmyrsrc_load(path)
Xchar *path;
X{
X  long fsize, lseek();
X  int rsc, size;
X  char *p;
X  extern char *pbase;
X
X  if((rsc = open(path, O_RDONLY | O_BINARY)) < 0) {
X    fprintf(stderr, "%s: Can't open '%s'.\n", progname, path);
X    return(0);
X  }
X
X  if((fsize = lseek(rsc, 0L, 2)) < 0)
X    return(0);
X
```

```
X  lseek(rsc, 0L, 0);
X
X  if(fsize < sizeof(RSHDR)) {
X    fprintf(stderr, "%s: Resource file is too small.\n", progname);
X    close(rsc);
X    return(0);
X  }
X
X  p = pbase = (char *)(Malloc(fsize));
X  if(pbase == NULL) {
X    fprintf(stderr, "%s: Not enough memory.\n", progname);
X    close(rsc);
X    return(0);
X  }
X
X  do {
X    size = (fsize >= BLKSIZE) ? BLKSIZE : (int)(fsize);
X    if(read(rsc, p, size) != size) {
X      fprintf(stderr, "%s: Read error on '$s'.\n", progname, path);
X      Mfree(pbase);
X      close(rsc);
X      return(0);
X    }
X    p += size;
X    fsize -= (long)(size);
X  } while(fsize > 0L);
X
X  close(rsc);
X
X  fix_pointers();
X
X  return(1);
X}
X
X
X/*-----------------------------------------*/
X/* Routine to fix up the resource pointers. */
X/*-----------------------------------------*/
X
Xfix_pointers()
X{
X  int i;
X  long *t;
X  extern char *pbase;
X  extern OBJECT *objects, *trindex[];
X  extern RSHDR *rschead;
X
X  /*-----------------------------------------------------*/
X  /* Initialize the pointer to the resource file header */
X  /* and a pointer to the base of the object array.     */
X  /*-----------------------------------------------------*/
X
X  rschead = (RSHDR *)(pbase);
```

```
X  objects = (OBJECT *)(pbase + rschead->rsh_object);
X
X  t = (long *)(pbase + rschead->rsh_trindex); /* Get pointer to trindex */
X
X  for(i=0; i<rschead->rsh_ntree; i++)
X    trindex[i] = (OBJECT *)(pbase + *(t+i));  /* Build tree pointer */
X
X  /*------------------------------------------------*/
X  /* Fix all the objects.  No need to go into the maze */
X  /* of head, tail, next because all objects are in a  */
X  /* nice array of size 'rsh_nobs'.                    */
X  /*------------------------------------------------*/
X
X  for(i=0; i<rschead->rsh_nobs; i++)
X    fix_one_object(objects + i);
X}
X
X
X/*-------------------------------------*/
X/* Fix up the pointers for one object. */
X/*-------------------------------------*/
X
Xfix_one_object(g)
Xregister OBJECT *g;
X{
X  TEDINFO *t;
X  ICONBLK *icon;
X  BITBLK *bitblk;
X  extern char *pbase;
X
X  switch(g->ob_type) {
X    case(G_STRING):
X    case(G_TITLE):
X    case(G_BUTTON):
X      g->ob_spec = (char *)(pbase + (long)(g->ob_spec));
X      break;
X    case(G_FTEXT):
X    case(G_FBOXTEXT):
X    case(G_TEXT):
X    case(G_BOXTEXT):
X      t = (TEDINFO *)(pbase + (long)(g->ob_spec));
X      g->ob_spec = (char *)(t);
X      t->te_ptext = (char *)(pbase + (long)(t->te_ptext));
X      t->te_ptmplt = (char *)(pbase + (long)(t->te_ptmplt));
X      t->te_pvalid = (char *)(pbase + (long)(t->te_pvalid));
X      break;
X    case(G_ICON):
X      icon = (ICONBLK *)(pbase + (long)(g->ob_spec));
X      g->ob_spec = (char *)(icon);
X      icon->ib_pmask = (int *)(pbase + (long)(icon->ib_pmask));
X      icon->ib_pdata = (int *)(pbase + (long)(icon->ib_pdata));
X      icon->ib_ptext = (char *)(pbase + (long)(icon->ib_ptext));
X      break;
```

```
X    case(G_IMAGE):
X      bitblk = (BITBLK *)(pbase + (long)(g->ob_spec));
X      g->ob_spec = (char *)(bitblk);
X      bitblk->bi_pdata = (int *)(pbase + (long)(bitblk->bi_pdata));
X      break;
X  }
X}
X
X
X/*---------------------------------*/
X/* Print the rs_trindex[] comment. */
X/*---------------------------------*/
X
Xprint_index_comment(g)
XOBJECT *g;
X{
X  register int i;
X  extern OBJECT *trindex[];
X  extern FILE *fp;
X  extern RSHDR *rschead;
X
X  for(i=0; i<rschead->rsh_ntree; i++) {
X    if(trindex[i] == g)
X      fprintf(fp, "  /* Tree #%d*/\n", i);
X  }
X}
X
X/* END OF FILE */
X
SHAR_EOF
chmod 0600 MAIN.C || echo "restore of MAIN.C fails"
sed 's/^X//' << 'SHAR_EOF' > MAKEFILE &&
X#-------------------------------------------------------------------------------
X#           Makefile to make the program 'rsctoc.ttp'
X#-------------------------------------------------------------------------------
X
XOBJECT = rsctoc.ttp
XFILES1 = main.o
X
X$(OBJECT): $(FILES1)
X    $(CC) $(FILES1) -o $(OBJECT)
X
Xclean:
X    $(RM) $(FILES1)
X    $(RM) $(OBJECT)
X
Xarc:
X    arc a rsctoc main.c makefile rsctoc.doc readme
X
X#-------------------------------------------------------------------------------
SHAR_EOF
chmod 0600 MAKEFILE || echo "restore of MAKEFILE fails"
sed 's/^X//' << 'SHAR_EOF' > README &&
```

```
X
XRsctoc is a program to generate C code out of a resource file.  It has been
Xwritten to be compiled with the Megamax Laser C compiler.  I do not know
Xhow much work it would be to compile it with another compiler.
X
XThe makefile that I use will not be of much use.  I have provided it only
Xas a guide.
X
XBob Silliker
SHAR_EOF
chmod 0600 README || echo "restore of README fails"
sed 's/^X//' << 'SHAR_EOF' > RSCTOC.DOC &&
X    RSCTOC                         VERSION 1.0                          RSCTOC
X
X
X    NAME
X        rsctoc.ttp - a program to generate C source from a resource file.
X
X    AUTHOR
X        Bob Silliker
X
X    COPYRIGHT
X        Copyright (c) 1988 by Robert Joseph Silliker
X        All Rights Reserved.
X        Permission is granted for unrestricted non-commercial use.
X
X    SYNOPSIS
X        rsctoc.ttp rsc_file [-p prefix] [-o file]
X
X    DESCRIPTION
X        Rsctoc is a program to generate C code from a resource file on an
X        Atari St computer.  It will handle the following types; OBJECT,
X        TEDINFO, ICONBLK and BITBLK.  If the command is executed with only
X        a path to the rsc_file then the C code will be displayed on the
X        screen.  To send the output to a file you can use the '-o file'
X        option or to redirect the output with '>file'.
X
X        The C code that is generated includes a small routine to fixup the
X        objects by calling rsrc_obfix() on each object to convert the
X        object's location size from character coordinates to pixel coordinates.
X        The routine is that is called to fixup the objects is 'fix_objects()'
X        and should be called once during program initialization (calling it
X        more than once is interesting but not useful).
X
X        The -p prefix option is used to add a short unique prefix to the names
X        of the structures, routine and macros that rsctoc creates.  This is
X        very useful when creating library routines that have trees embedded
X        in them (providing a means of preventing name conflicts).
X
X        If rsctoc is executed in the following way
X
X                    rsctoc rsc_file -p my -o file
X
```

```
X        Then the C code generated would have 'my' added to the start of the
X        structures, routine and macros.
X
X                    rs_tedinfo      would be      myrs_tedinfo
X                    NUM_TI          would be      myNUM_TI
X                    fix_objects     would be      myfix_objects
X
X    NOTES
X        The program DOES NOT generate the arrays rs_strings[], rs_frstr[],
X        rsfrimg[], or rs_imdope[] arrays.  The program DOES generate the
X        arrays rs_tedinfo[], rs_object[], rs_trindex[], rs_bitblk[],
X        rs_iconblk[] arrays as well as defines the macros NUM_TI, NUMOBS,
X        NUM_TREE, NUM_BB and NUM_IB (modified by the prefix of course).
X
X        Any names that you assigned to objects, trees, images or icons can
X        still be used in your program to access these elements. The program
X        will only handle 512 trees (not objects, images or icons which are
X        only limited by the resource file structure) in one input resource
X        file.
X
X    BUGS
X        There are no known bugs in V1.0. (:->)
X        If you find one fix it.
SHAR_EOF
chmod 0600 RSCTOC.DOC || echo "restore of RSCTOC.DOC fails"
exit 0


From -4646308393079179881
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,35ca83cd7b1ab887,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-06-20 10:52:47 PST
Path: utzoo!attcan!uunet!husc6!bbn!uwmcsd1!ig!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i058: fastdump -- text-only screen dump
Keywords: Lattice
Message-ID: <3828@saturn.ucsc.edu>
Date: 20 Jun 88 17:52:47 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 374
Approved: koreth@ssyx.ucsc.edu

Submitted-by: uunet!mcvax!philmds!leo (Leo de Wit)
Posting-number: Volume 1, Issue 58
Archive-name: fastdump

another submission for the ST. It runs in any case on a color monitor
(med and high res); I have no chance of testing it on a monochrome
monitor.

The following program (FASTDUMP.PRG) should be placed in the AUTO
```

folder, so that it is made resident at startup time. After pressing
Alternate-Help the screen is dumped to the printer (at least, a copy of
the graphic image 8-). It is much faster than the standard routine but
only suitable for text screens.

The program was compiled and linked with the compiler and linker that
come with the Lattice-C compiler, as follows:

    Click on LC.TTP, then enter:
fastdump

    Create a file fastdump.lnk with contents (2 lines):
INPUT fastdump.bin
LIBRARY clib.bin

    Click on LINK.TTP, then enter:
-with fastdump

The resulting code was 1924 bytes in size.

    L.J.M. de Wit
    Nachtegaallaan 7
    5731XP Mierlo
    Holland

---------- H E R E I T A L L S T A R T S ----------

```
/*
 ****************************************************************************
 *                                                                          *
 *    fastdump.c  version 1.1 of 19 June 1988    (C) L.J.M. de Wit 1988     *
 *                                                                          *
 * This software may be used and distributed freely if not used commercially  *
 * and the originator (me) is mentioned.                                    *
 *                                                                          *
 ****************************************************************************
 *
 * NAME
 *    fastdump - fast ascii screen dump
 *
 * SYNTAX
 *    fastdump.prg
 *
 * DESCRIPTION
 *    Fastdump does a fast screen dump whenever the Alt Help key is pressed.
 *    It reads the screen memory to match on font characters; matched
 *    characters are printed as-is, non-matched characters as spaces.
 *    It has some intelligence as to avoid printing trailing spaces.
 *    It is very useful as afterwards printer, e.g. after a compilation with
 *    error messages, or a program that exits with failure messages, and also
 *    for printing source/text from within an editor, or for programs that do
 *    not provide for hardcopy output.
 *
```

```
 *    Fastdump should be in the AUTO folder as FASTDUMP.PRG so that it is
 *    installed memory resident when the system is loaded. It then overrides
 *    the standard screen dump routine.
 *
 * BUGS
 *    The program does not provide for RS232 printing yet. The Centronics port
 *    is used.
 *
 *    The routine is not used when it is called from the XBIOS. This is because
 *    of old disk-based TOS versions that had the dump vector hardcoded. A
 *    future release of this program (if there will be any &-) will use the
 *    screen dump vector at address 0x502.
 *
 *    Characters in reverse video are not matched yet with their reverse images
 *    so that reverse video now results in spaces. Also something for the next
 *    release.
 *
 * DECISIONS
 *    The current version is compiled with a Lattice C compiler; people using
 *    other compilers should be aware of the fact that on entrance of do_dump()
 *    the contents of A4/A5 will be different from what it was when the
 *    program was started (being an interrupt-routine) so that register
 *    relative addressing cannot be used in this case (or A4 should be loaded
 *    when the routine is entered). For Lattice C absolute addressing is the
 *    default so that there was no problem in this case.
 *    After compilation the module should be linked only with the standard
 *    library (for the gemdos function). So the link file should contain:
 *            INPUT fastdump.bin
 *            LIBRARY clib.bin
 *    startup.bin is not used, and the function startup() is used as entry point.
 *    The size of the resulting code is much smaller now (not using stdio etc).
 *
 */

#include <osbind.h>
#include <portab.h>

#define DUMPFLG  (*(WORD *)0x4EE)            /* Alt Help pressed flag       */
#define SSHIFTMD (*(BYTE *)0x44C)            /* Screen Shift Mode           */
#define V_BAS_AD (*(LONG *)0x44E)            /* logical screen base address */
#define NVBLS    (*(WORD *)0x454)            /* length of this array        */
#define VBLQUEUE (*(que_ptr *)0x456)         /* array of VBL vectors        */
#define HZ_200   (*(LONG *)0x4BA)            /* 200 Hz system clock         */
#define CENTRONICS_BUSY (1 & *(BYTE *)0xFFFFFA01)
#define GI       ((BYTE *)0xFFFF8800)        /* I/O locations for sound chip*/

/* font definitions */
typedef struct font_header {
   WORD f_id;       /* font identifier; system font = 1 */
   WORD f_point;    /* font size in points */
   char f_name[32]; /* font name */
   WORD f_low;      /* lowest ascii code in font */
   WORD f_high;     /* highest ascii code in font */
```

```
   WORD f_top;      /* relative distance of top to base line */
   WORD f_ascent;   /* relative distance of ascent to base line */
   WORD f_half;     /* relative distance of half to base line */
   WORD f_descent;  /* relative distance of descent to base line */
   WORD f_bottom;   /* relative distance of bottom to base line */
   WORD f_maxcharw; /* maximal character width in font */
   WORD f_maxcellw; /* maximal cell width in font */
   WORD f_loffset;  /* left offset */
   WORD f_roffset;  /* right offset */
   WORD f_fatsiz;   /* degree of fattening */
   WORD f_ulsiz;    /* underline degree */
   WORD f_normask;  /* normal cancelled mask */
   WORD f_skewmask; /* skewing cancelled mask */
   WORD f_flag;     /* flag:
                     *  bit 0: system font;
                     *  bit 1: hor. offset;
                     *  bit 2: byte swap;
                     *  bit 3: non proportional font
                     */
   WORD *f_horoff;  /* pointer to horizontal offsets table */
   WORD *f_charoff; /* pointer to character offsets table */
   char *f_data;    /* pointer to font data table */
   WORD f_fontw;    /* total width of all chars in font */
   WORD f_fonth;    /* height of font (# scanlines) */
   struct font_header
          *f_next;  /* pointer to next font header */
} font_header;

static UWORD g_fheader[] = {  0xA000,        /* LineA exception 0   */
                              0x222F,0x0004, /* MOVE.L  4(SP),D1    */
                              0xE581,        /* ASL.L   #2,D1       */
                              0x2031,0x1800, /* MOVE.L  0(A1,D1),D0 */
                              0x4E75 };      /* RTS                 */

static UWORD get_sr[]    = {  0x7000,        /* MOVEQ.L #0,D0       */
                              0x40C0,        /* MOVE.W  SR,D0       */
                              0x007C,0x0700, /* OR.W    #$700,SR    */
                              0x4E75 };      /* RTS                 */

static UWORD set_sr[]    = {  0x46EF,0x0006, /* MOVE.W 6(SP),SR     */
                              0x4E75 };      /* RTS                 */

WORD planes;
font_header *fhp;

static void do_dump(), startup();
static BYTE lstout(),  get_schar(),
            read_gi(), write_gi(), dummy();

typedef void (**que_ptr)();  /* que_ptr points to an array of pointers
                              *    to functions returning void
                              */
```

```
static void startup(base)
LONG *base;
{
   LONG memneed;
   extern LONG _mneed;
   WORD i, nvbls;
   LONG ssp;

   memneed = 0x100   +                    /* base page   */
             base[3] +                    /* text length */
             base[5] +                    /* data length */
             base[7] +                    /* bss  length */
             0x800   +                    /* workspace   */
             0x900;                       /* for stack   */

   ssp = Super(0);                        /* Supervisor mode             */
   nvbls = NVBLS;                         /* length of VBLQUEUE array    */

   for (i = 0; i < nvbls; i++) {
      if (VBLQUEUE[i] == (void (*)())0) {/* If empty slot found          */
         VBLQUEUE[i] = do_dump;          /* set vector for new routine   */
         break;
      }
   }

   Super(ssp);                            /* Back to User mode again     */

   if (i < nvbls) {                       /* If empty slot was found     */
      Ptermres(memneed,0);                /* make resident & terminate   */
   } else {                               /* else report the problem &   */
      Cconws("Cannot bind new VBL routine\r\n");
      Pterm(1);                           /* terminate with error status */
   }
}

static void do_dump()                     /* The interrupt routine       */
{
   WORD i, j;
   char *s, *t;
   WORD rows, cols;                       /* Rows & columns of screen    */
   char line[81];                         /* Holds a line to be printed  */
   BYTE status;                           /* Printer ready status        */
   BYTE rez;                              /* resolution 2 high 1 med 0 low*/

   if (DUMPFLG != 0) {                    /* Alt Help not pressed?       */
      return;
   }

   if ((rez = SSHIFTMD & 3) == 3) {
      rez = 2;
   }
   fhp = (*(font_header *(*)())g_fheader)
         ((rez == 2) ? 2 : 1);            /* pointer to standard font    */
```

```
   planes = 4 >> rez;                     /* # of bit planes: 4, 2 or 1  */
   rows = 25;
   cols = (rez == 0) ? 40 : 80;

   for (i = 0; i < rows; i++) {           /* Handle each screen row ...  */
      s = line; t = line;
      for (j = 0; j < cols; j++, s++) {   /* Handle each column per row  */
         *s = get_schar(i,j);             /* Get ASCII value at this pos.*/
         if ((*s != ' ') || (*t != ' ')) {/* t: last nonsp. or first sp. */
            t = s;
         }
      }
      *s = '\0';                          /* null-terminate line         */
      if (*t == ' ') {
         *t = '\0';                       /* discards trailing spaces    */
      }

      status = 0;
      for (s = line; *s != '\0'; s++) {   /* print each char/test status */
         status = lstout(*s);
         if (status != 0) break;          /* printer not ready; abort    */
      }

      if (status != 0) break;             /*        abort                */

      lstout('\r'); lstout('\n');         /* terminate line with CR/LF   */

      if (DUMPFLG != 0) {                 /* Alt Help pressed again?     */
         break;
      }
   }

   DUMPFLG = -1;                          /* No Hardcopy                 */
}

static BYTE lstout(c)                     /* Send one character          */
char c;                                   /* to Centronics port          */
{
   LONG start;
   BYTE not_ready;

   start = HZ_200;
   do {                                   /* poll the status             */
      not_ready = CENTRONICS_BUSY;
      dummy();                            /* to prevent erroneous optim. */
   } while (not_ready &&
               (HZ_200 - start < 6000)); /* 30 sec. timeout             */

   if (!not_ready) {                      /* The actual printing         */
      BYTE val;
      LONG istatus;

      istatus = (*(LONG (*)())get_sr)();  /* Save SR                     */
```

```
      val = read_gi(7);
      write_gi(7,val | 0x80);
      (*(LONG (*)())set_sr)(istatus);     /* Restore SR                  */
      write_gi(15,c);

      val = read_gi(14);
      write_gi(14,val & 0xDF);
      val = read_gi(14);
      write_gi(14,val | 0x20);
   }

   return not_ready;                      /* return 0 for OK             */
}

static BYTE read_gi(reg)                  /* Read from G.I. sound chip   */
BYTE reg;
{
   LONG istatus;

   istatus = (*(LONG (*)())get_sr)();     /* Save SR                     */
   GI[0] = reg;
   reg = dummy();                         /* prevents optimizing away .. */
   reg = GI[0];                           /* ... this statement          */
   (*(LONG (*)())set_sr)(istatus);        /* Restore SR                  */

   return reg;
}

static BYTE dummy()                       /* This function is used to    */
{                                         /* fool the compiler so that   */
   return 0;                              /* no statement is optimized   */
}                                         /* away (volatile variables)   */

static BYTE write_gi(reg,val)             /* Write to G.I. sound chip    */
BYTE reg,val;
{
   LONG istatus;

   istatus = (*(LONG (*)())get_sr)();     /* Save SR                     */
   GI[0] = reg;
   GI[2] = val;
   reg = dummy();                         /* prevents optimizing away .. */
   reg = GI[0];                           /* ... this statement          */
   (*(LONG (*)())set_sr)(istatus);        /* Restore SR                  */

   return reg;
}

static BYTE get_schar(i,j)                /* Find match in a font for    */
WORD i,j;                                 /* the character bit image at  */
{                                         /* position (row,col)          */
   register WORD l;                       /* line counter                */
   register BYTE *cpd,                    /* char ptr into font_data     */
```

```
                *cps;                   /* char ptr into ch_img[]      */
   register WORD p, c;
   WORD *curadd,                        /* screen address of top word  */
        sval;                           /* will hold a word of image   */
   BYTE ch_img[16],                     /* will hold image as bytes    */
        or_all;                         /* OR of all bytes of image    */
   UWORD w_p_l,                         /* # words per line            */
         maxl;                          /* height of a char in lines   */

   w_p_l = (planes == 1) ? 40 : 80;
   maxl = (planes == 1) ? 16 : 8;
   curadd = (WORD *)(V_BAS_AD + i * 1280 + (j & ~1) * planes);

   /* prepare ch_img[] to hold a adjusted copy of the bit image */
   or_all = 0;
   for (l = 0; l < maxl; l++) {
      sval = 0;
      for (p = 0; p < planes; p++) {    /* OR in all colour bit planes */
         sval |= curadd[l * w_p_l + p];
      }

      ch_img[l] = (j & 1) ? sval & 0xff   /* Take lower or upper byte    */
                  : (sval >> 8) & 0xff;   /* as appropriate              */
      or_all |= ch_img[l];              /* Keeps inclusive Or of all   */
   }

   /* search */
   if (or_all == 0) {                   /* Not a pixel set             */
      c = ' ';                          /* then space will be printed  */
   } else {                             /* else for each char in font  */
      for (c = fhp->f_low; c <= fhp->f_high; c++) {
         cpd = fhp->f_data + (fhp->f_charoff[c] >> 3);
         cps = ch_img;
         for (l = 0; l < maxl; l++) {   /* Compare each line (byte)    */
            if (*cps++ != *cpd) break;  /* Match failed at this line   */
            cpd += fhp->f_fontw;
         }
         if (l >= maxl) break;          /* All lines matched           */
      }

      if ((c == '\0') || (c > fhp->f_high)) { /* If no match           */
         c = ' ';                       /* use space instead           */
      }
   }

   return (BYTE)c;                      /* Return matched char or space*/
}


From 4042146580808967828
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,a15ca427d0a4db79,start
X-Google-Attributes: gid100493,public
```

X-Google-ArrivalTime: 1988-06-22 11:02:10 PST
Path: utzoo!utgpu!water!watmath!clyde!att!pacbell!ames!pasteur!agate!saturn!ssyx!
koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i059: readme12 -- online help system
Keywords: MWC
Message-ID: <3885@saturn.ucsc.edu>
Date: 22 Jun 88 18:02:10 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 974
Approved: koreth@ssyx.ucsc.edu

Submitted-by: ucbvax!ihnp4!stag!trb (Todd Burkey)
Posting-number: Volume 1, Issue 59
Archive-name: readme12

The following is the source code and executable for a README
application that implements my new POPHELP mechanism. Full source
code is included for README (MW C) and it runs in both Medium Res
(RGB) and High Res (mono). A menu configuration file that describes
POPHELP is also included (and automatically presented to the user
when they run README.PRG).

The program is self documenting (it would be silly if it wasn't).
My primary goal in writing the POPHELP routines was to beef up
HDSCAN's interactive help and to lay the groundwork for full popup
menu control for users of HDSCAN. It ended up to be such short and
simple code that I decided to put it out for others to use, add to
their own programs, or simply learn some coding techniques from.
The code has been implemented in the next version of HDSCAN (took
about 2 hours to integrate it) and does tend to snaz up your application
some. One of the local developers has even created a README pophelp
configuration file for Mark Johnson C. Enjoy.

Some features of POPHELP:

1) External Help file driven (think of it as a help resource file).
   Can also be fully integrated into the application code.
2) Allows context sensitive help applications to be easily written.
3) Users can wander through the help popups via arrow keys or the
   mouse.
4) The popups are very fast and consist of selectable and
   non-selectable text in shadowed popup boxes.
5) The implementor can vary screen color from popup to popup.
6) External files can be 'fed' into a popup menu (saving on memory
   for little used/accessed help topics).

Some features of README:

1) Originally intended as a test vehicle for checking out the
   POPHELP routines, I actually am using README to replace my README
   files for HDSCAN and DISKSCAN. Much cleaner user interface.
2) Allows a user defined 'front page' from which the HELP key can be
   pressed to allow a user to 'learn' about an application in a
   logical fashion.

```
   -Todd Burkey     "A member of STdNET - The ST developers' Network"
    trb@stag.UUCP

------------------------------------------------------------
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      README12.C
#      MENU.TRB
#      README.TRB
#      SAMPLE.TXT
#
sed 's/^X//' << 'SHAR_EOF' > README12.C &&
X               /* README.PRG Vers. 1.0 By Todd Burkey, 6/1/88. */
X/* Version 1.1 6/10/88-Todd Burkey: Added Color selection and Tab support */
X/* Version 1.2 6/19/88-Todd Burkey: Started code setup for menu overlays  */
X
X/* The source code and accompany text files (MENU.TRB and README.TRB) must
X   always accompany the README.PRG file (ARC'ed as a package).             */
X
X/*
X   This program shows one use of the POPHELP mechanism (Copyright 1988 by
X   Todd Burkey). POPHELP is intended to be a 'clean' way of presenting help
X   to users of a program and was also written to demonstrate in source code
X   how to do those confusing linea/vdi/etc things that all of our manuals kind
X   of gloss over. I am not saying that my methods are proper coding or even
X   close, but at least it works. In order to ensure that this code and the
X   techniques involved stay accessible to the public at large, I have to make
X   the following stipulations:
X
X     1) This program and source code may be freely distributed as long as no
X        profits are made in the process (Club Disks of the Month and BBS's are
X        excepted, of course.)
X     2) All derivatives of the routines in this code must include this notice
X        verbatim and are similarly subject to the conditions in 1, above.
X     3) No warrantees are made as to the usefulness or safety of using this
X        code. The user uses it at his/her own risk.
X
X         -Todd Burkey    "A member of STdNET-The ST developers' Network"
X          trb@stag.UUCP
X          3546 Pilgrim Lane
X          Plymouth, MN 55441
X*/
X
X#include <linea.h>
X#include <ctype.h>
X#include <stdio.h>
X#include <aesbind.h>
X#include <osbind.h>
X#include <vdibind.h>
X#include <bios.h>
```

```
X
X#define RESTORE 1
X#define SAVE 2
X#define ONLYTEXT 1
X#define MOREHELP 2
X#define ESC 27
X#define HELP 98
X#define UPARROW 72
X#define DOWNARROW 80
X#define LEFTARROW 75
X#define RIGHTARROW 77
X#define LEFTBUTTON 0x740000L
X#define RIGHTBUTTON 0x750000L
X#define MAX_MENUS 100
X
X/* and now some necessary vdi/aes stuff */
Xint h,xy[16],contrl[12],intin[128],ptsin[128],intout[128],ptsout[128];
Xint src[11];
Xint scr_a[] = {  0, 0, 640, 200, 40, 0, 2, 0, 0, 0 };
Xchar *oldscr,*newscr,*memblk;
X
X/* and now for the POPMENU related arrays */
Xint menu_col[MAX_MENUS+1],     /* upper left column number for menu */
X    menu_row[MAX_MENUS+1],     /* upper left row number for menu */
X    menu_fr[MAX_MENUS],        /* first character in menu_s for the menu */
X    menu_to[MAX_MENUS],        /* last character in menu_s for the menu */
X    menu_last[MAX_MENUS],      /* previous menu that you viewed */
X    on_line[MAX_MENUS],        /* what line were you last on? */
X    menu_rgb[MAX_MENUS+1][4];  /* color rgb map for menu */
Xchar menu_s[30000];
Xint error_cnt=0;
Xint which;                     /* which menu are we on */
Xint menu_length,               /* length (height) of current menu */
X    menu_width,                /* width of current menu */
X    lc1,lc2,lr1,lr2,           /* last corners (col/row) of menu box */
X    menu_ptr[24],              /* a menu selections' next menu (forward ptr) */
X    menu_type[24];             /* type of menu entry...ONLYTEXT or MOREHELP */
Xchar menu_line[24][130];       /* what you see in the box */
Xchar help_path[100];           /* tbd...for global path lookup... */
Xint menu_s_pos;                /* tbd...useful if we use menu overlays */
X
X/* and some stuff I need */
Xlong tst,vtst;
Xint htst,mono,black,blue,red,green;
Xint save_col[4];
Xlong _stksize = 40000L;
Xstatic long patmsk = -1;
X
Xmain()
X{
X  int i;
X  color_init();
X  ws_init();             /* do all that messy vdi/linea initialization */
```

```
X  build_help();          /* build the internal help arrays from menu.trb */
X  if(Bconstat(BC_KBD)) Bconout(BC_KBD,0x0a);  /* set mouse cursor mode */
X  if(Bconstat(BC_KBD)) Bconout(BC_KBD,0x14);  /* 20 pulses per x move */
X  if(Bconstat(BC_KBD)) Bconout(BC_KBD,0x05);  /* 5 pulses per y move */
X  mon_keys();
X  if(Bconstat(BC_KBD)) Bconout(BC_KBD,0x08);  /* reset mouse mode to normal */
X  for(i=0;i<4;i++) Setcolor(i,save_col[i]);
X  }
X
Xcolor_init()   /* check resolutions and setup accordingly */
X{
X  int i;
X  i=Getrez();
X  if(i==0) {
X    printf("Sorry, Medium or High Res only\n[Hit any key]\n");
X    Crawcin();
X    exit(1);
X    }
X  if(i==2) {
X     mono=1;
X     black='3';
X     blue='3';
X     red='3';
X     green='0';
X     }
X  else {
X    mono=0;
X    black='2';
X    blue='0';
X    red='1';
X    green='2';
X    }
X  for(i=0;i<4;i++) save_col[i]=Setcolor(i,-1);
X  for(i=0;i<=MAX_MENUS;i++) {          /* setup default colors */
X    if(mono) menu_rgb[i][0]=0x777;
X    else     menu_rgb[i][0]=0x006;
X    menu_rgb[i][1]=0x600;
X    menu_rgb[i][2]=0x000;
X    if(mono) menu_rgb[i][3]=0x000;
X    else     menu_rgb[i][3]=0x775;
X    }
X  set_rgb(0);
X  }
X
Xmon_keys()   /* simple loop that waits for a HELP key or Q key */
X{
X  while(((tst=Bconin(2))&0xFF)!='Q') {
X    tst=tst>>16;
X    if(tst==HELP) {
X      which=0;
X      show_help(1);
X      }
X    while((Cconis())!=0) Crawcin();
```

```
X    }
X  v_clsvwk(h);
X  appl_exit();
X  }
X
Xset_rgb(m)
Xint m;
X{
X  Setcolor(0,menu_rgb[m][0]);
X  Setcolor(1,menu_rgb[m][1]);
X  Setcolor(2,menu_rgb[m][2]);
X  Setcolor(3,menu_rgb[m][3]);
X  }
X
Xdraw_help(menu)  /* draw the selected help popup */
Xint menu;
X{
X  int i,j;
X/* First, gather up all of the info for the CURRENT menu to draw */
X  menu_length=0;
X  if(menu>=MAX_MENUS||menu<0)
X    sprintf(menu_line[menu_length++],"ERROR IN MENU.TRB. Menu numbers must be
between 0 and 99.");
X  else if(menu_fr[menu]<0)
X    sprintf(menu_line[menu_length++],"ERROR IN MENU.TRB. Can't Find the menu
definition for %d.",menu);
X  if(menu_length>0) {
X    sprintf(menu_line[menu_length++],"Please fix this problem ASAP. Press the
Left Arrow now.");
X    menu_width=strlen(menu_line[0]);
X    menu_type[0]=menu_type[1]=ONLYTEXT;
X    menu=MAX_MENUS;
X    menu_col[menu]=4;
X    menu_row[menu]=11;
X    }
X  else if(menu_s[menu_fr[menu]]!='<') {
X    for(i=menu_fr[menu],j=0,menu_width=0;i<=menu_to[menu];i++) {
X      if(menu_s[i]=='\0') {
X        menu_line[menu_length++][j-1]='\0';
X        j++;
X        if(j>menu_width) menu_width=j;
X        j=0;
X        }
X      else {
X        if(j==0&&menu_s[i]=='+') menu_type[menu_length]=MOREHELP;
X        else if(j==0&&menu_s[i]=='T') menu_type[menu_length]=ONLYTEXT;
X        else if(j==0&&menu_s[i]=='=') {
X          menu_ptr[menu_length-1]=atoi(&menu_s[i+2]);
X          while(menu_s[i]!='\0'&&i<10000) i++;
X          j=0;
X          }
X        else if(j<=1&&menu_s[i]==':') j++;
X        else {
```

```
X          menu_line[menu_length][j-1]=menu_s[i];
X          j++;
X          }
X        }
X      }
X    menu_width=menu_width-2;
X    }
X  else viewfile(menu);
X  set_rgb(menu);
X  lc1=menu_col[menu];
X  lr1=menu_row[menu];
X  lc2=lc1+menu_width;
X  lr2=lr1+menu_length;
X/* Now, save the part of the screen that we are gonna draw on. */
X  screen_it(SAVE,lc1,lr1,lc2,lr2);
X/* and draw the popup */
X  fboxcr(lc1,lr1,lc2,lr2);
X  boxcr(lc1,lr1,lc2,lr2);
X  for(i=0;i<menu_length;i++) {
X    if(menu_type[i]==ONLYTEXT) {
X      if(mono) printcr(lc1,lr1+i,menu_line[i]); /* modify for slant someday */
X      else    tprintcr(lc1,lr1+i,menu_line[i]);
X      }
X    else {
X      if(mono) printcr(lc1,lr1+i,menu_line[i]);
X      else bprintcr(lc1,lr1+i,menu_line[i]);
X      }
X    }
X  }
X
Xerase_help()   /* move the stuff we saved to the scratch pad back */
X{
X  screen_it(RESTORE,lc1,lr1,lc2,lr2);
X  }
X
Xhilite_menu(i)   /* highlight the selection under the cursor */
Xint i;
X{
X  if(mono) {
X    printcr(lc1,lr1+on_line[which],menu_line[on_line[which]]);
X    bprintcr(lc1,lr1+i,menu_line[i]);
X    }
X  else {
X    bprintcr(lc1,lr1+on_line[which],menu_line[on_line[which]]);
X    printcr(lc1,lr1+i,menu_line[i]);
X    }
X  on_line[which]=i;
X  }
X
Xshow_help(menu)  /* recursive routine handling the help popup walking */
Xint menu;
X{
X  int i,all_text;
```

```
X  if(which<0) return;
X  draw_help(menu);
X  on_line[which]=0;
X  for(i=0;i<menu_length&&menu_type[i]==ONLYTEXT;i++);
X  if(i<menu_length) {
X    all_text=0;
X    on_line[which]=i;
X    if(mono) bprintcr(lc1,lr1+i,menu_line[i]);
X    else      printcr(lc1,lr1+i,menu_line[i]);
X    }
X  else all_text=1;
X  while(((tst=Bconin(2))&0xFF)!=ESC) {
X    htst=tst&0xFF;
X    vtst=tst>>16;
X    if(!all_text&&which<100&&(tst==RIGHTBUTTON||vtst==RIGHTARROW)) {
X      erase_help();
X      menu_last[which++]=menu;
X      show_help(menu_ptr[on_line[which-1]]);
X      if(which<0) return;
X      continue;
X      }
X    else if(which>0&&(tst==LEFTBUTTON||vtst==LEFTARROW)) {
X      if(menu>=0) {
X        erase_help();
X        menu=menu_last[--which];
X        draw_help(menu);
X        if(menu_type[on_line[which]]!=ONLYTEXT)
X          if(mono) bprintcr(lc1,lr1+on_line[which],menu_line[on_line[which]]);
X          else      printcr(lc1,lr1+on_line[which],menu_line[on_line[which]]);
X        return;
X        }
X      }
X    else if(on_line[which]<menu_length&&(vtst==DOWNARROW||htst=='j')) {
X      for(i=on_line[which]+1;i<menu_length&&menu_type[i]==ONLYTEXT;i++);
X      if(i<menu_length) hilite_menu(i);
X      }
X    else if(on_line[which]>0&&(vtst==UPARROW||htst=='k')) {
X      for(i=on_line[which]-1;i>0&&menu_type[i]==ONLYTEXT;i--);
X      if(menu_type[i]!=ONLYTEXT) hilite_menu(i);
X      }
X    else {
X      if(menu!=0&&vtst!=UPARROW&&vtst!=DOWNARROW&&vtst!=LEFTARROW&&vtst!
=RIGHTARROW)
X        error_cnt++;
X      if(error_cnt>1||vtst==HELP) {
X        erase_help();
X        error_cnt=0;
X        menu_last[which++]=menu;
X        show_help(0);
X        if(which<0) return;
X        continue;
X        }
X      }
```

```
X    while((Cconis())!=0) Crawcin();
X    }
X  erase_help();
X  which= -1;
X  }
X
Xpop_read(s) /* This routine reads a menu file and builds the internal tree */
Xchar *s;
X{
X  char buf[128];
X  FILE *ifp;
X  int i,j,x,y,r,g,b,on_menu;
X  if((ifp=fopen(s,"r"))==NULL) {
X    printf("menu configuration file:%s was not found!\n",s);
X    exit(99);
X    }
X  j=menu_s_pos;                /* pointer into menu_s array */
X  if(ifp!=NULL) {
X    while(fgets(buf,128,ifp)!=NULL) {
X      if(buf[1]!=':') continue;
X      switch(buf[0]) {
X        case '<':
X        case 'T':
X        case '+':
X        case '=':
X          for(i=0;i<strlen(buf);i++)
X            if(buf[i]!='\t') menu_s[j++]=buf[i];
X            else {
X              x=expand_tab(j);
X              for(y=j;y<x;y++) menu_s[j++]=' ';
X              }
X          menu_s[j-1]='\0';
X          break;
X        case '#':
X          if(j>0) menu_to[on_menu]=j-1;
X          on_menu=atoi(&buf[2]);
X          if(on_menu>MAX_MENUS) {
X            printf("ERROR. Can't define menu. Its' number [%d] is
>100...aborting\n",on_menu);
X            exit(1);
X            }
X          menu_fr[on_menu]=j;
X          break;
X        case 'X':
X          menu_col[on_menu]=atoi(&buf[2]);
X          break;
X        case 'Y':
X          menu_row[on_menu]=atoi(&buf[2]);
X          break;
X        case '0':
X        case '1':
X        case '2':
X        case '3':
```

```
X          if(strlen(buf)<6) printf("ERROR. RGB menu definition too short:
%s\n",buf);
X          r=buf[2]-'0';
X          g=buf[3]-'0';
X          b=buf[4]-'0';
X          if(r<0||r>7||g<0||g>7||b<0||b>7)
X            printf("ERROR. RGB values incorrect (must be 0-7): %s\n",buf);
X          if(on_menu==1)
X            for(i=1;i<=MAX_MENUS;i++) menu_rgb[i][buf[0]-'0']=r*256+g*16+b;
X          else menu_rgb[on_menu][buf[0]-'0']=r*256+g*16+b;
X          break;
X        }
X      }
X    if(j>0) menu_to[on_menu]=j-1;
X    fclose(ifp);
X    }
X  }
X
Xbuild_help() /* this routine reads the menu file and builds the popup tree */
X{
X  int i;
X  for(i=0;i<MAX_MENUS;i++) menu_fr[i]= -1;
X  menu_s_pos=0;                /* pointer into menu_s array */
X  pop_read("MENU.TRB");
X  }
X
Xviewfile(menu)  /* gather menu information from a file */
Xint menu;       /* would be nice to modify this to allow paging... :-) */
X{              /* Note that we build a fake menu structure from the file */
X  int i,j,x,y;
X  char ebuf[150],buf[129],s[200];
X  FILE *ifp;
X  strcpy(s,&menu_s[menu_fr[menu]+2]);
X  if ((ifp=fopen(s,"r"))==NULL) {   /* throw up a read-only one-liner */
X    sprintf(menu_line[0],"Sorry, can't open %s. Press the left arrow.",s);
X    menu_length=1;
X    menu_width=strlen(menu_line[0]);
X    menu_type[0]=ONLYTEXT;
X    return;
X    }
X  menu_length=0;
X  menu_width=0;
X  while((menu_length+menu_row[menu])<24&&fgets(buf,128,ifp)!=NULL) {
X    j=0;
X    for(i=0;i<strlen(buf);i++)
X      if(buf[i]!='\t') ebuf[j++]=buf[i];
X      else {
X        x=expand_tab(j);
X        for(y=j;y<x;y++) ebuf[j++]=' ';
X        }
X    ebuf[j-1]='\0';
X    strcpy(menu_line[menu_length],ebuf);
X    menu_type[menu_length++]=ONLYTEXT;
```

```
X    if(j>menu_width) menu_width=j;
X    }
X  fclose(ifp);
X  }
X
Xws_init()
X{
X   int ii,i,work_in[11],work_out[57];
X   linea0();             /* yep, we are going to try linea stuff */
X   src[0]=0;src[1]=0;    /* general setup for doing vdi stuff follows*/
X   work_in[0]=1;
X   for(i=1;i<10;i++) work_in[i]=1;
X   work_in[10]=2;
X   appl_init();
X   h=graf_handle(&ii,&ii,&ii,&ii);
X   graf_mouse(256,NULL);
X   clear_screen();
X   v_opnvwk(work_in,&h,work_out);
X   src[2]=work_out[0]+1;
X   src[3]=work_out[1]+1;
X   src[4]=src[2]>>4;
X   src[5]=0;
X   if(mono) src[6]=1;
X   else     src[6]=2;
X   if(mono) {        /* modify fdb for the vro_cpyfm */
X     scr_a[3]=400;   /* 400 pixels high */
X     scr_a[6]=1;     /* 1 bit plane */
X     }
X   memblk=(char *)Malloc(32*1024L);                    /* alloc 32k */
X   oldscr=(char *) Physbase();                         /* where is screen */
X   newscr=(char *) (((long) memblk+0xFFL)& ~(0xFFL));    /* assign paste */
X   *((long *)src)=(long) oldscr;     /* assign ptr to screen */
X   *((long *)scr_a)=(long) newscr;   /* assign ptr to paste */
X   v_hide_c(h);
X   draw_cover();
X   }
X
Xclear_screen()
X{
X  v_clrwk(h);
X  }
X
Xdraw_cover()  /* display a setup (cover) page */
X{
X  char buf[129];
X  FILE *ifp;
X  int i;
X  i=0;
X  clear_screen();
X  if((ifp=fopen("README.TRB","r"))==NULL)
X    printf("README.TRB not found! If MENU.TRB exists, press HELP for more info.
\n");
X  else {
```

```
X    printf("\33f\33Y%c%c\33pReady? Press The HELP key for more info, then ESC and
Q to quit\33q",56,42);
X    while(fgets(buf,128,ifp)!=NULL&&i<24) {
X      if(buf[0]=='/') continue;
X      if(i==0) printf("\33Y  %s",buf);
X      else    printf("%s",buf);
X      i++;
X      }
X    fclose(ifp);
X    }
X  }
X
Xtprintcr(c,r,string)  /* print a red string at <col,row> */
Xint c,r;
Xchar *string;
X{
X  r=r+32;
X  c=c+32;
X  printf("\33Y%c%c\33c%c\33p%s\33q\33c%c\n",r,c,red,string,blue);
X  }
X
Xbprintcr(c,r,string)  /* print a 'bold/reverse video' string at <col,row> */
Xint c,r;
Xchar *string;
X{
X  r=r+32;
X  c=c+32;
X  printf("\33Y%c%c\33p%s\33q\n",r,c,string);
X  }
X
Xprintcr(c,r,string)  /* print a string at <column,row> */
Xint c,r;
Xchar *string;
X{
X  r=r+32;
X  c=c+32;
X  printf("\33Y%c%c%s\n",r,c,string);
X  }
X
Xfboxcr(c1,r1,c2,r2)  /* draw a shadow box and then an overlay one. */
Xint c1,r1,c2,r2;     /* using linea for the fun of it */
X{
X  LNMASK= -1;
X  WMODE= 0;
X  CLIP=0;
X  PATPTR=&patmsk;
X  PATMSK=1;
X  if(mono) COLBIT0=1;
X  else {
X    COLBIT0=0;
X    COLBIT1=1;
X    }
X  X1=(c1<<3)+5;
```

```
X  X2=(c2<<3)+7;
X  if(mono) {
X    Y1=(r1<<4)-10;
X    Y2=(r2<<4)-6;
X    }
X  else {
X    Y1=(r1<<3)-5;
X    Y2=(r2<<3)-3;
X    }
X  linea5();     /* lot of work for a stupid rectangle, but this shows linea */
X  if(mono) COLBIT0=0;
X  else     COLBIT0=1;
X  X1=(c1<<3)-2;
X  X2=(c2<<3);
X  if(mono) {
X    Y1=(r1<<4)-2;
X    Y2=(r2<<4);
X    }
X  else {
X    Y1=(r1<<3)-1;
X    Y2=(r2<<3);
X    }
X  linea5();
X  boxcr(c1,r1,c2,r2);
X  }
X
Xboxcr(c1,r1,c2,r2)   /* now draw a hollow box around the top filled...*/
Xint c1,r1,c2,r2;
X{
X  xy[0]=xy[6]=xy[8]=(c1<<3)-3;
X  xy[2]=xy[4]=(c2<<3)+1;
X  if(mono==1) {
X    xy[1]=xy[3]=xy[9]=(r1<<4)-3;
X    xy[5]=xy[7]=(r2<<4)+1;
X    }
X  else {
X    xy[1]=xy[3]=xy[9]=(r1<<3)-2;
X    xy[5]=xy[7]=(r2<<3)+1;
X    }
X  v_pline(h,5,xy);
X  }
X
Xscreen_it(wh,c1,r1,c2,r2)  /* do a simple blit to or from the paste */
Xint wh,c1,r1,c2,r2;        /* columns and rows in cursor coordinates */
X{
X  int lx,ux,ly,uy;
X  lx=(c1<<3)-4;
X  ux=(c2<<3)+7;
X  if(mono==1) {
X    ly=(r1<<4) - 16;
X    uy=(r2<<4)+4;
X    if(uy>399) uy=399;
X    }
```

```
X  else {
X    ly=(r1<<3) - 8;
X    uy=(r2<<3)+2;
X    if(uy>199) uy=199;
X    }
X  if(lx<0) lx=0;
X  if(ly<0) ly=0;
X  if(ux>639) ux=639;
X  xy[1]=xy[5]=ly;      /* uly */
X  xy[3]=xy[7]=uy;      /* lry */
X  xy[0]=xy[4]=lx;      /* ulx */
X  xy[2]=xy[6]=ux;      /* lrx */
X  if(wh==SAVE)
X    vro_cpyfm(h,3,xy,src,scr_a);
X  else
X    vro_cpyfm(h,3,xy,scr_a,src);
X  }
X
Xint expand_tab(index)       /* expand a tab out, calculating new position */
Xint index;
X{
X  do
X    index++;
X  while ((index&0x07)!=0);
X  return(index);
X  }
X
SHAR_EOF
chmod 0600 README12.C || echo "restore of README12.C fails"
sed 's/^X//' << 'SHAR_EOF' > MENU.TRB &&
X*      SAMPLE MENU TREE FOR POPHELP (Todd Burkey/5-30-88)      */
X/* menu for level 0 help */
X#:0
XX:20
XY:2
XT:    POPHELP By Todd Burkey
XT:
XT:Uh-oh. You need HELP for POPHELP?
XT:OK, the first thing to remember is
XT:that in POPHELP, you really only
XT:need to use the four arrow keys
XT:and the ESC key. The up and down
XT:arrows will move you up and down in
XT:the current pop-up...but only when
XT:there is selectable (blue) text in
XT:the popup. Note that this popup has
XT:NO selectable text...you can just
XT:read it and exit. Which brings us
XT:to the left and right arrow keys.
XT:The right key gets you into a menu,
XT:while the left key moves you back
XT:out of a menu. The ESC key? It gets
XT:you out of the popups altogether.
```

```
XT:You can press the LEFT ARROW now.
X#:1
XX:5
XY:15
XT:WELCOME TO POPHELP
X+:HELP ON POPHELP
X=:3
X+:POPHELP MANIFESTO
X=:40
X+:Who made this mess
X=:41
X+:ABOUT README
X=:50
X+:Some Error Testing
X=:42
X+:Other menu #4
X=:4
X+:Other Menu #6
X=:6
X
X#:50
XX:4
XY:4
X1:530
XT:README is a generic application that I hope you find useful in setting
XT:up useful on-line documentation for programs you develop. Feel free to
XT:give out copies of README to others who are learning C or who you feel
XT:could use a better help mechanism. I will try to keep the pophelp file
XT:formats upwards compatible, so keep your eyes open for future releases
XT:of improved versions of this program.
X
X#:40
XX:2
XY:4
XT:This program shows one use of the POPHELP mechanism (Copyright 1988 by
XT:Todd Burkey). POPHELP is intended to be a 'clean' way of presenting help
XT:to users of a program and was also written to demonstrate in source code
XT:how to do those confusing linea/vdi/etc things that all of our books kind
XT:of gloss over. I am not saying that my methods are proper coding or even
XT:close, but at least it works. In order to ensure that this code and the
XT:techniques involved stay accessible to the public at large, I have to
XT:make the following stipulations:
XT:
XT: 1) This program and source code may be freely distributed as long as no
XT:    profits are made in the process (Club Disks of the Month and BBSs are
XT:    excepted, of course.)
XT: 2) All derivatives of the routines in this code must include this notice
XT:    verbatim and are similarly subject to the conditions in 1, above.
XT: 3) No warrantees are made as to the usefulness or safety of using this
XT:    code. The user uses it at his/her own risk.
X
X#:41
X1:030
```

```
X2:540
XX:14
XY:15
XT: Todd Burkey                 A member of STdNET
XT: 3546 Pilgrim Lane        The ST developers' Network
XT: Plymouth, MN 55441            (trb@stag.UUCP)
X
X#:42
XX:10
XY:6
XT:  SIMPLE TESTING
X+:Look at sample.txt
X=:10
X+:Look at bad file
X=:11
X+:Open Menu >100
X=:120
X+:Menu Doesn't exist
X=:88
X+:Goto myself (loop)
X=:42
X
X/* Real help for POPHELP */
X#:3
XX:12
XY:8
XT:GETTING STARTED
X+:Moving Around
X=:22
X+:Some Comments
X=:23
X+:Technical Info
X=:26
X
X#:26
XX:30
XY:10
X+:MENU.TRB Format
X=:27
X+:COLOR Tables
X=:28
X
X#:27
XX:3
XY:4
XT:The file MENU.TRB is processed by POPHELP's build_menu routine and
XT:drives the entire menu structure of POPHELP. The format of MENU.TRB
XT:is very simple. For example, the following is a sample 2 level menu
XT:with comments (note the comments can't really be part of the line):
XT:#:1             <-we are defining menu # 1
XT:X:10            <-the popup box will start at cursor column 10
XT:Y:4             <-and at cursor row number 4
XT:T:WELCOME       <-just some text to put in red (color #1)
```

```
XT:+:More Help     <-text for another menu selection
XT:=:2             <-go to menu 2 when the above text is selected
XT:#:2             <-all done with menu 1, so lets define menu 2 now
XT:X:5             <-new coordinates
XT:Y:6             <-new coordinates
XT:1:070           <-redefine color 1 (the text) to be green (RGB)
XT:<:afile.txt     <-menu 2 popup text is read from a file
X
X#:60
XX:20
XY:8
X1:530
X2:053
XT:For RGB users only
XT:  You were warned
X+:Some soft colors
X=:61
X+:Some Putrid ones
X=:62
X+:Late night colors
X=:63
X
X#:61
XX:5
XY:3
X0:054
X1:410
X2:046
X3:222
XT:Pastels?
XT:0:054
XT:1:500
XT:2:066
XT:3:333
X
X#:62
XX:5
XY:3
X0:070
X1:770
X2:077
X3:707
XT:Putrid
XT:0:070
XT:1:770
XT:2:077
XT:3:707
X
X#:63
XX:5
XY:3
X0:000
X1:700
```

```
X2:007
X3:770
XT:Brighter
XT:0:000
XT:1:700
XT:2:007
XT:3:770
X
X#:28
XX:2
XY:4
XT:Colors can be changed from the default setup dynamically within the menus,
XT:but extreme care should be taken so you don't screw things up for people
XT:who use monochrome monitors. The technique is quite simple. Within the
XT:MENU.TRB file, include the following fields when you want a new color:
XT:0:006          <--change color 0 to light blue (normal background)
XT:1:600          <--change color 1 to red (non-selectable text)
XT:2:000          <--change color 2 to black (for the shadow)
XT:3:775          <--change color 3 to off white (popup background).
XT:Note that the default color can be over-ridden by specifying a new color
XT:for menu # 1 (the first popup that is displayed by POPHELP).
X+:Some sample colors maps.
X=:60
X
X#:22
XX:14
XY:10
X+:ARROW KEYS
X=:24
X+:HOW TO QUIT
X=:25
X
X#:25
XX:40
XY:12
XT:1) Press the ESC key.  This takes
XT:   you out of the help selection.
XT:2) Press the Q (not q) key. This
XT:   exits you from the demo.
X
X#:24
XX:30
XY:12
XT:RIGHT ARROW = Select the hilited menu
XT:LEFT ARROW  = Move to menu we came from
XT:UP ARROW    = Move up in the menu list
XT:DOWN ARROW  = Move down in the list
X
X#:23
XX:15
XY:11
XT:POPHELP is a simple, but clean, mechanism
XT:for both externalized and internally held
```

```
XT:help information. It was written only for
XT:fun, but may prove useful in standardized
XT:help utilities as its' features are tuned
XT:up and refined.  Any comments from anyone
XT:out there? Or is this idea just too wierd
XT:and far-fetched.
X
X/* lower level menu that is all text */
X#:4
XX:30
XY:14
XT:This is all Text, so you
XT:can't really do much but
XT:look at it and then push
XT:the left arrow key.
X
X/* small menu */
X#:6
XX:50
XY:3
XT:Menu 6
X+:GOTO 5
X=:5
X+:GOTO 1
X=:1
X
X/* menu that is really a file */
X#:10
XX:3
XY:5
X<:sample.txt
X
X/* just a test to show what a missing file will do */
X#:11
XX:3
XY:5
X<:hl.txt
X
X/* another little text menu */
X#:5
XX:70
XY:20
XT:JUST
XT:MORE
XT:Text
SHAR_EOF
chmod 0600 MENU.TRB || echo "restore of MENU.TRB fails"
sed 's/^X//' << 'SHAR_EOF' > README.TRB &&
X/* This isn't meant to be read THIS way! Run README.PRG and check it out.    */
X/* This file MUST be in the same directory as README.PRG for README to run.  */
X/* Since you are looking at this file, note that this is the file that makes */
X/* up the beginning display when you first invoke README.PRG. By replacing   */
X/* This page with your own, and making your own MENU.TRB file, you will have */
```

```
X/* your own customized standalone help reader */
X
X                            POPUP HELP MENU DEMO
X                               by Todd Burkey
X
X  Hate clicking on and paging through README files?  Well, give this program a
X  try. README.PRG is a standalone program that uses my POPUP HELP mechanism to
X  create nice, heirarchical help menus. Originally intended for HDSCAN, I have
X  decided to release this README application w/source code so others out there
X  can use it, learn from it, and hopefully even improve on it. POPHELP is easy
X  to use, just press HELP and use the arrows to move around.  Press HELP again
X  (while inside POPHELP) if you get confused.  Yep, even mono-chrome and mouse
X  lovers are supported. Some technical notes:
X
X    1) POPHELP is file driven. A procedure reads in MENU.TRB which describes
X       the popup hierarchy. Think of this as a help resource file.
X    2) POPHELP has a limit of 100 popup menus (as compiled herein). Menu 0 is
X       for people who get confused.
X    3) Short (<24 line) files can be automatically shown (see the examples.)
SHAR_EOF
chmod 0600 README.TRB || echo "restore of README.TRB fails"
sed 's/^X//' << 'SHAR_EOF' > SAMPLE.TXT &&
X/* SAMPLE Two level Tree */
X#:1
XX:5
XY:15
XT:WELCOME TO POPHELP
X+:Some Error Testing
X=:42
X
X#:42
XX:10
XY:6
XT:LEVEL TWO MENU
X+:Goto menu 1
X=:1
X+:Menu Doesn't exist
X=:88
SHAR_EOF
chmod 0600 SAMPLE.TXT || echo "restore of SAMPLE.TXT fails"
exit 0


From 6439595159499834157
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,4e576bd86bd93e8d,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-06-26 15:48:33 PST
Path: utzoo!attcan!uunet!husc6!cmcl2!nrl-cmf!ames!pasteur!ucbvax!agate!saturn!
ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i060: nlqps -- near-letter-quality printer driver
```

Keywords: Lattice
Message-ID: <3932@saturn.ucsc.edu>
Date: 26 Jun 88 22:48:33 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 530
Approved: koreth@ssyx.ucsc.edu

Submitted-by: uunet!mcvax!philmds!leo (Leo de Wit)
Posting-number: Volume 1, Issue 60
Archive-name: nlqps

Another nice program for printer owners. That is, if your printer does
not have NLQ capability, but does have double-density graphics. It is,
at least I think it is, as good as NLQ in hardware; besides you have
the possibility of proportional spacing.
    The program consists of 3 modules: newstart.c, nlqps.c and
nlqpsdat.c and a header file: nqlps.h. It is compiled and linked with
Lattice C, but it should be easy to adapt it for other compilers. The
printer it was tested with was a Star Gemini-10X.
   Generation procedure for Lattice: compile all three .c files, then
link them, together with the C library, to form a program named
nlqps.ttp. The correct linking procedure : create a link file,
nlqps.lnk, that contains:

```
INPUT newstart.bin
INPUT nlqps.bin
INPUT nlqpsdat.bin
LIBRARY clib.bin
```

and then give the command:
```
LINK.TTP -WITH NLQPS -NOLIST -PROG NLQPS.TTP
```

   For a manual page I refer to the first comment in nlqps.c

    L.J.M. de Wit
    Nachtegaallaan 7
    5731XP Mierlo
    Holland

    E-mail: ...!mcvax!philmds!leo

```
================== SHAR ARCHIVE ========== CUT HERE ========================
: This is a shar archive.  Extract with sh, not csh.
: This archive ends with exit, so do not worry about trailing junk.
echo 'Extracting newstart.c'
sed 's/^X//' > newstart.c << '+ END-OF-FILE newstart.c'
X/*
X ***************************************************************************
X *                                                                         *
X *    newstart.c  version 1.1 of 18 Feb  1988    (C) L.J.M. de Wit 1988     *
X *                                                                         *
X * This software may be used and distributed freely if not used commercially  *
X * and the originator (me) is mentioned.                                   *
```

```
X *                                                                            *
X ******************************************************************************
X *
X * This module (the first to be linked in) may only be used when malloc and/or
X * stdio are not used.
X * Startup() is the entry point which frees memory and resets the stack.
X * Then startetc() is called which initiates argc & argv for main; they
X * could not be on start's stack because that is reset.
X * exit() is redefined to avoid linking it from the library.
X */
X
X#include <osbind.h>
X
X#define TRUE         1
X#define FALSE        0
X#define ISSPACE(c) (((c) == ' ') || ((c) == '\t') || ((c) == '\n'))
X#define ACNT 32
X
Xtypedef unsigned char bool;
X
Xchar **environ;
X
Xstatic void startup(), startetc();
Xextern int main();
Xvoid exit();
X
Xstatic void startup(base)
Xchar *base;
X{
X   int memneed;
X   static short setsp[] = { /* set stack to value supplied */
X      0x2057,      /* MOVEA.L  (SP),A0  */
X      0x2E6F, 4,   /* MOVEA.L 4(SP),SP  */
X      0x4ED0       /* JMP      (A0)     */
X   };
X   extern int _mneed;
X
X   memneed = 0x100 +            /* base page   */
X             ((int *)base)[3] + /* text length */
X             ((int *)base)[5] + /* data length */
X             ((int *)base)[7] + /* bss  length */
X             0x800 +            /* workspace   */
X             0x900;             /* for stack   */
X
X   if (_mneed > memneed) {    /* take more memory */
X      memneed = (char *)(&memneed) - base;
X      if (_mneed < memneed) { /* but nomore as requested */
X         memneed = _mneed;
X      }
X   }
X
X   Mshrink(base,memneed);              /* frees mem: takes what's needed */
X   (*(void (*)())setsp)(base+memneed); /* resets sp to base+memneed */
```

```
X   startetc(base);                    /* to also have correct frame pointer */
X}
X
Xstatic void startetc(base)
Xchar *base;
X{
X   int argc, i;
X   char *argv[ACNT], *envp[ACNT];
X   char evstr[256], *s, *t;
X   bool inarg;             /* whether in a argument */
X
X   if ((s = ((char **)base)[11]) == (char *)0) {
X      *evstr = '\0';
X   } else {
X      for (t = evstr; *t++ = *s++; ) ;
X   }
X
X   /* stack arguments */
X   argv[0] = (char *)0; inarg = FALSE;
X   for (argc = 1, i = 0; (argc < ACNT) && (i < (unsigned)base[128]); i++) {
X      if ((base[129+i] == '\0') || ISSPACE(base[129+i])) {
X         if (inarg) {
X            inarg = FALSE;
X            base[129+i] = '\0';
X         }
X      } else {
X         if (!inarg) {
X            inarg = TRUE;
X            argv[argc++] = base+129+i;
X         }
X      }
X   }
X   base[129+i] = '\0';
X
X   if (argc == ACNT) {
X      Cconws("newstart: too many arguments\r\n");
X      exit(1);
X   }
X
X   inarg = FALSE;
X   for (i = 0, s = evstr; (i < ACNT) && (*s != '\0'); s++) {
X      if (ISSPACE(*s)) {
X         if (inarg) {
X            inarg = FALSE;
X            *s = '\0';
X         }
X      } else {
X         if (!inarg) {
X            inarg = TRUE;
X            envp[i++] = s;
X         }
X      }
X   }
```

```
X   if (i == ACNT) {
X      Cconws("newstart: too many environment variables\r\n");
X      exit(1);
X   }
X
X   environ = envp;
X
X   argv[argc] = (char *)0;
X   envp[i]    = (char *)0;
X   main(argc,argv,envp);
X   exit(0);
X}
X
Xvoid exit(code)
Xint code;
X{
X   Pterm(code);
X}
+ END-OF-FILE newstart.c
chmod 'u=rw,g=r,o=' 'newstart.c'
echo 'SENT: -rw-r-----  1 leo          3688 Jun 24 12:30 newstart.c'
echo -n 'RCVD: '
/bin/ls -l newstart.c
echo 'Extracting nlqps.c'
sed 's/^X//' > nlqps.c << '+ END-OF-FILE nlqps.c'
X/*
X ****************************************************************************
X *                                                                          *
X *    nlqps.c     version 1.1 of 24 June 1988    (C) L.J.M. de Wit 1988     *
X *                                                                          *
X * This software may be used and distributed freely if not used commercially  *
X * and the originator (me) is mentioned.                                    *
X *                                                                          *
X ****************************************************************************
X *
X * NAME
X *    nlqps - print with Near Letter Quality and Proportional Spacing
X *
X * SYNTAX
X *    nlqps.ttp [-n|-N] [file ...]
X *
X * DESCRIPTION
X *    This program is meant for use with printers that have no NLQ capability.
X *    The printer SHOULD have double density graphics capability and capability
X *    of sending a LF of half a scanline high (1/144 inch). The Star Gemini-10X
X *    is an example. If the escape sequences for your printer are different,
X *    you should alter the macros LP_SMALL_LF() (1/144 inch linefeed),
X *    LP_BIG_LF() (23/144 inch linefeed) and LP_DD_GRAPH() (double density
X *    graphics) accordingly.
X *
X *    The program prints your file(s) (or standard input for pipes) in a
X *    standard NLQ fashion, i.e. the line is printed twice in graphics mode
X *    with complementary fonts and a very small linefeed inbetween. The total
```

```
X *    linefeed is still 1/144 + 23/144 = 1/6 inch, which is a standard
X *    line height.
X *
X *    A nice feature of this program is that it does proportional spacing as
X *    well; if this is not desired (e.g. when printing tables, columns, program
X *    listings) this feature can be turned off by the -n switch; n for
X *    non-proportional (-N is the same and only needed because silly uppercase
X *    conversion of the parameter line when running TTP's from the Desktop).
X *
X * BUGS/SHORTCOMINGS
X *    The program is not activated when printing files from the Desktop. The
X *    question is: Is automatic NLQ printing desired (printing without NLQ
X *    becomes harder) ?
X *
X *    The program does not yet provide for different NLQ fonts. This is however
X *    not hard (loading a file into the nlqfont struct) but only useful with
X *    an editor program for such a font.
X *
X * DECISIONS
X *    For compactness of code the module should be linked with newstart.bin
X *    instead of startup.bin (the standard first module and entry point of the
X *    program). This new startup module avoids using stdio (not needed here),
X *    resulting in a much smaller program: 4435 bytes instead of 14036!
X *    The standard C library IS used (for the gemdos/bios interface).
X *    One should create a link file, nlqps.lnk, that contains:
X *            INPUT newstart.bin
X *            INPUT nlqps.bin
X *            INPUT nlqpsdat.bin
X *            LIBRARY clib.bin
X *    and the command to be given is:
X *    LINK.TTP -WITH NLQPS -NOLIST -PROG NLQPS.TTP
X *    The function startup() in newstart.c is used as entry point, so
X *    newstart.bin must be the first module to be linked in.
X */
X
X#include <osbind.h>
X#include <portab.h>
X#include "nlqps.h"
X
X#define BUFSIZ     512  /* from stdio.h */
X
X#define SET0         0  /* denotes set to be used in second scan */
X#define SET1         1  /* denotes set to be used in first scan */
X#define INTERSPACE   3  /* space between characters in dot columns */
X#define ESC         27  /* escape character */
X#define BUFLEN     256  /* max size of character buffer for one line */
X#define MAXSCAN    960  /* max # of dot columns per line */
X#define CHARWIDTH   12  /* width of char in dot columns if non-proportional */
X
X#define LP(c) Bconout(0,(c))
X#define LP_SMALL_LF() (LP(ESC),LP(74),LP(1))
X#define LP_BIG_LF()   (LP(ESC),LP(74),LP(23))
X#define LP_DD_GRAPH() (LP(ESC),LP('L'))
```

```
X
Xstatic void prline(), nlq();
Xstatic int lookup();
X
Xstatic int notprop;
X
Xint main(argc,argv)
Xint argc;
Xchar **argv;
X{
X   int i, fd;
X
X   i = 1;
X
X   notprop = (argc >= 2)                  /* Check argv[1] for "-n" or "-N" */
X             && (argv[i][0] == '-')
X             && ((argv[i][1] == 'n') || (argv[i][1] == 'N'));
X   if (notprop) i++;
X
X   if (i == argc) {
X      nlq(0);                             /* Nlq the standard input         */
X   } else {
X      for ( ; i < argc; i++) {
X         if ((fd = Fopen(argv[i],0)) < 0) {
X            Cconws("nlqps: "); Cconws(argv[i]); Cconws(": cannot open\r\n");
X         } else {
X            nlq(fd);                      /* Nlq the file                   */
X            Fclose(fd);
X         }
X      }
X   }
X}
X
Xstatic void nlq(fd)                       /* Print from filedescriptor fd   */
Xint fd;                                   /* in NLQ with prop. spacing      */
X{
X   char inbuf[BUFSIZ],                   /* Input buffer for reads from fd */
X        *ibp = inbuf,                    /* For reading a char from inbuf  */
X        c;
X   char buf[BUFLEN],                     /* Line to print (still ASCII)    */
X        *dummy;
X   int bufcnt = 0,                       /* # chars in buf sofar           */
X       scancnt = 0,                      /* # dot columns sofar            */
X       nread = 0;                        /* # bytes read into inbuf        */
X
X   do {
X      if (inbuf + nread == ibp) {         /* Read all chars from inbuf      */
X         if ((nread = Fread(fd,sizeof(inbuf),inbuf)) <= 0) { /* Fill inbuf  */
X            break;                        /* End-of-file                    */
X         }
X         ibp = inbuf;                     /* Pointer at buffer start        */
X      }
X      c = *ibp++;                         /* Read 1 char from inbuf         */
```

```
X      if ((c == '\n')                      /* Had newline                    */
X         || (bufcnt == BUFLEN)             /* .. or max # characters         */
X         || (scancnt > MAXSCAN - 9)) {     /* .. or max # dot columns        */
X
X         if (scancnt > 0) {                /* Do first scan                  */
X            LP_DD_GRAPH();                 /* Select dd graphics             */
X            LP(scancnt & 0xff);            /* # dot columns: low byte        */
X            LP((scancnt >> 8) & 0xff);     /* # dot columns: high byte       */
X            prline(buf,bufcnt,SET1);       /* Print buf graphic with SET1    */
X         }
X         LP_SMALL_LF();                    /* Send a 1/144 inch LF           */
X         if (scancnt > 0) {                /* Do second scan                 */
X            LP_DD_GRAPH();
X            LP(scancnt & 0xff);
X            LP((scancnt >> 8) & 0xff);
X            prline(buf,bufcnt,SET0);       /* Print buf graphic with SET0    */
X         }
X         LP_BIG_LF();                      /* Send a 23/144 inch LF          */
X         bufcnt = scancnt = 0;             /* Reset buf parameters           */
X      }
X      if (c != '\n') {                     /* normal character               */
X         buf[bufcnt++] = c;                /* Put it into buf; update bufcnt */
X         scancnt += (notprop)              /* .. and scancnt:                */
X            ? CHARWIDTH                    /* constant CHARWIDTH if not prop.*/
X            : lookup(c,SET0,&dummy)        /* real character width + space   */
X              + INTERSPACE;                /* If proportional                */
X      }
X   } while (TRUE);
X}
X
Xstatic void prline(buf,bufcnt,set)         /* Print the buf (bufcnt chars)   */
Xchar *buf;                                  /* with this set in graphic mode  */
Xint bufcnt, set;
X{
X   int i, j,
X       scancnt;                            /* width of char in dot columns   */
X   char *ch_start;                         /* start of char bit image (9)    */
X
X   for (i = 0; i < bufcnt; i++) {          /* For each char in buf           */
X      scancnt = lookup(buf[i],set,&ch_start);/* set width and start         */
X      for (j = 0; j < scancnt; j++) {      /* Do each dot column             */
X         LP(ch_start[j]);
X      }
X      for (j = (notprop) ? CHARWIDTH - scancnt : INTERSPACE; j > 0; --j) {
X         LP(0);                            /* Print empty column as spacing  */
X      }
X   }
X}
X
Xstatic int lookup(c,set,csp)               /* Get byte image start & # dot   */
Xchar c, **csp;                              /* columns for this char and set  */
Xint set;
X{
```

```
X   int retval;
X
X   c -= ' ';
X   if ((c >= 0) && (c < 128 - ' ')) {
X      retval = nlqfont[c].nlq_width;
X      *csp   = (set == SET0) ? nlqfont[c].nlq_set0 : nlqfont[c].nlq_set1;
X   } else {                                /* If c out of range             */
X      retval = 0;
X      *csp = "";
X   }
X   return retval;                          /* actual width of image         */
X}
+ END-OF-FILE nlqps.c
chmod 'u=rw,g=r,o=' 'nlqps.c'
echo 'SENT: -rw-r-----  1 leo          8759 Jun 24 12:41 nlqps.c'
echo -n 'RCVD: '
/bin/ls -l nlqps.c
echo 'Extracting nlqps.h'
sed 's/^X//' > nlqps.h << '+ END-OF-FILE nlqps.h'
X/*
X ****************************************************************************
X *                                                                          *
X *    nlqps.h     version 1.1 of 24 June 1988    (C) L.J.M. de Wit 1988     *
X *                                                                          *
X * This software may be used and distributed freely if not used commercially  *
X * and the originator (me) is mentioned.                                    *
X *                                                                          *
X ****************************************************************************
X *
X * nlqps.h: typedef & external declaration for nlqps
X */
X
Xtypedef struct nlqchar {
X   unsigned char nlq_width;
X   char nlq_set0[9];
X   char nlq_set1[9];
X} nlqchar;
X
Xextern nlqchar nlqfont[];
+ END-OF-FILE nlqps.h
chmod 'u=rw,g=r,o=' 'nlqps.h'
echo 'SENT: -rw-r-----  1 leo           837 Jun 24 12:28 nlqps.h'
echo -n 'RCVD: '
/bin/ls -l nlqps.h
echo 'Extracting nlqpsdat.c'
sed 's/^X//' > nlqpsdat.c << '+ END-OF-FILE nlqpsdat.c'
X/*
X ****************************************************************************
X *                                                                          *
X *    nlqpsdat.c  version 1.1 of 24 June 1988    (C) L.J.M. de Wit 1988     *
X *                                                                          *
X * This software may be used and distributed freely if not used commercially  *
X * and the originator (me) is mentioned.                                    *
```

```
X *                                                                            *
X ******************************************************************************
X *
X *    nlqpsdat.c: This file contains the (initialized) structure for the
X *                nlqps font.
X */
X
X#include "nlqps.h"
X
Xnlqchar nlqfont[] = {
X/* ' ' */ {6, 0,0,0,0,0,0,0,0,0,            0,0,0,0,0,0,0,0,0},
X/* '!' */ {1, 224,0,0,0,0,0,0,0,0,          242,0,0,0,0,0,0,0,0},
X/* '"' */ {5, 192,0,0,0,192,0,0,0,0,        224,0,0,0,224,0,0,0,0},
X/* '#' */ {9, 0,0,252,0,0,0,252,0,0,        40,40,254,40,40,40,254,40,40},
X/* '$' */ {9, 96,0,0,0,252,0,0,0,24,        32,84,84,84,254,84,84,84,8},
X/* '%' */ {9, 128,0,132,8,16,32,68,128,4,   192,194,196,8,16,32,70,134,6},
X/* '&' */ {9, 200,20,32,16,192,12,0,12,0,   76,160,146,162,74,0,4,0,10},
X/* ''' */ {3, 0,0,192,0,0,0,0,0,0,          160,160,192,0,0,0,0,0,0},
X/* '(' */ {3, 48,204,0,0,0,0,0,0,0,         56,68,130,0,0,0,0,0,0},
X/* ')' */ {3, 0,204,48,0,0,0,0,0,0,         130,68,56,0,0,0,0,0,0},
X/* '*' */ {7, 72,0,48,252,48,0,72,0,0,      68,40,16,254,16,40,68,0,0},
X/* '+' */ {9, 0,0,0,0,120,0,0,0,0,          16,16,16,16,124,16,16,16,16},
X/* ',' */ {3, 5,5,6,0,0,0,0,0,0,            6,6,7,0,0,0,0,0,0},
X/* '-' */ {9, 0,0,0,0,0,0,0,0,0,            16,16,16,16,16,16,16,16,16},
X/* '.' */ {4, 6,6,6,6,0,0,0,0,0,            2,2,2,2,0,0,0,0,0},
X/* '/' */ {7, 0,4,8,16,32,64,128,0,0,       2,4,8,16,32,64,128,0,0},
X/* '0' */ {9, 48,204,0,0,0,0,0,204,48,      56,68,130,130,130,130,130,68,56},
X/* '1' */ {5, 0,128,252,0,0,0,0,0,0,        66,66,254,2,2,0,0,0,0},
X/* '2' */ {9, 0,132,0,8,0,16,0,160,64,      66,2,134,130,138,130,146,2,98},
X/* '3' */ {9, 0,4,0,0,0,96,0,148,8,
132,128,130,130,162,130,210,128,140},
X/* '4' */ {9, 0,16,32,64,128,0,252,0,0,     8,24,40,72,136,136,254,8,8},
X/* '5' */ {9, 192,4,0,0,0,0,0,36,24,
228,160,162,162,162,162,162,128,156},
X/* '6' */ {9, 8,20,32,64,128,0,0,20,8,      12,16,50,82,146,18,18,0,12},
X/* '7' */ {9, 0,0,0,4,8,16,32,64,128,
128,128,130,132,136,144,160,192,128},
X/* '8' */ {9, 72,180,0,0,0,0,0,180,72,      108,0,146,146,146,146,146,0,108},
X/* '9' */ {9, 64,160,0,0,4,8,16,160,64,     96,0,144,144,146,148,152,16,96},
X/* ':' */ {4, 36,36,36,36,0,0,0,0,0,        54,54,54,54,0,0,0,0,0},
X/* ';' */ {4, 37,37,37,38,0,0,0,0,0,        54,54,54,55,0,0,0,0,0},
X/* '<' */ {7, 0,48,0,72,0,132,0,0,0,        16,0,40,0,68,0,130,0,0},
X/* '=' */ {9, 0,0,0,0,0,0,0,0,0,            40,40,40,40,40,40,40,40,40},
X/* '>' */ {7, 0,132,0,72,0,48,0,0,0,        130,0,68,0,40,0,16,0,0},
X/* '?' */ {9, 0,128,0,0,0,16,0,160,64,      64,0,128,128,138,138,144,0,96},
X/* '@' */ {9, 120,132,0,48,0,0,0,128,112,   124,0,130,146,170,170,170,42,114},
X/* 'A' */ {9, 28,32,64,128,0,128,64,32,28,  30,40,72,136,136,136,72,40,30},
X/* 'B' */ {9, 0,252,0,0,0,0,0,180,72,       130,254,146,146,146,146,146,0,108},
X/* 'C' */ {9, 120,132,0,0,0,0,0,132,0,      124,0,130,130,130,130,130,0,68},
X/* 'D' */ {9, 0,252,0,0,0,0,0,132,120,      130,254,130,130,130,130,130,68,56},
X/* 'E' */ {9, 252,0,0,0,0,0,0,0,0,
254,146,146,146,146,146,146,130,130},
X/* 'F' */ {9, 252,0,0,0,0,0,0,0,0,
```

```
254,144,144,144,144,144,144,128,128},
X/* 'G' */ {9, 120,132,0,0,0,0,0,4,24,        124,0,130,130,146,146,146,144,28},
X/* 'H' */ {9, 252,0,0,0,0,0,0,0,252,         254,16,16,16,16,16,16,16,254},
X/* 'I' */ {5, 0,0,252,0,0,0,0,0,0,           130,130,254,130,130,0,0,0,0},
X/* 'J' */ {8, 8,4,0,0,0,4,248,0,0,           12,0,2,130,130,128,252,128,0},
X/* 'K' */ {9, 252,0,0,48,0,72,0,132,0,       254,16,16,0,40,0,68,0,130},
X/* 'L' */ {9, 252,0,0,0,0,0,0,0,0,           254,2,2,2,2,2,2,2,2},
X/* 'M' */ {9, 252,128,64,32,0,32,64,128,252, 254,128,64,32,16,32,64,128,254},
X/* 'N' */ {9, 252,0,128,64,32,16,8,4,252,    254,128,64,32,16,8,4,2,254},
X/* 'O' */ {9, 120,132,0,0,0,0,0,132,120,     124,0,130,130,130,130,130,0,124},
X/* 'P' */ {9, 252,0,0,0,0,0,0,160,64,        254,144,144,144,144,144,144,0,96},
X/* 'Q' */ {9, 120,132,0,0,0,8,0,132,120,     124,0,130,130,138,130,134,0,126},
X/* 'R' */ {9, 252,0,0,0,0,16,8,164,64,       254,144,144,144,144,144,152,4,98},
X/* 'S' */ {9, 64,164,0,0,0,0,0,148,8,        100,0,146,146,146,146,146,0,76},
X/* 'T' */ {9, 0,0,0,0,252,0,0,0,0,
128,128,128,128,254,128,128,128,128},
X/* 'U' */ {9, 248,4,0,0,0,0,0,4,248,         252,0,2,2,2,2,2,0,252},
X/* 'V' */ {9, 224,16,8,4,0,4,8,16,224,       224,16,8,4,2,4,8,16,224},
X/* 'W' */ {9, 252,0,4,8,16,8,4,0,252,        252,2,4,8,16,8,4,2,252},
X/* 'X' */ {7, 132,72,48,0,48,72,132,0,0,     130,68,40,16,40,68,130,0,0},
X/* 'Y' */ {9, 128,64,32,16,12,16,32,64,128,  128,64,32,16,14,16,32,64,128},
X/* 'Z' */ {7, 0,4,8,16,32,64,128,0,0,        130,134,138,146,162,194,130,0,0},
X/* '[' */ {5, 252,0,0,0,0,0,0,0,0,           254,130,130,130,130,0,0,0,0},
X/* '\' */ {7, 128,64,32,16,8,4,0,0,0,        128,64,32,16,8,4,2,0,0},
X/* ']' */ {5, 0,0,0,0,252,0,0,0,0,           130,130,130,130,254,0,0,0,0},
X/* '^' */ {9, 0,64,0,128,0,128,0,64,0,       32,0,64,0,128,0,64,0,32},
X/* '_' */ {9, 1,1,1,1,1,1,1,1,1,             0,0,0,0,0,0,0,0,0},
X/* '`' */ {4, 128,64,0,0,0,0,0,0,0,          192,128,160,160,0,0,0,0,0},
X/* 'a' */ {9, 0,12,0,0,0,0,32,28,0,          4,0,42,42,42,42,10,28,2},
X/* 'b' */ {8, 252,0,0,0,0,0,36,24,0,         254,34,34,34,34,34,0,28,0},
X/* 'c' */ {8, 24,36,0,0,0,0,0,0,0,           28,0,34,34,34,34,34,34,0},
X/* 'd' */ {8, 24,36,0,0,0,0,0,252,0,         28,0,34,34,34,34,34,254,0},
X/* 'e' */ {9, 24,36,0,0,0,0,0,32,16,         28,8,42,42,42,42,42,10,24},
X/* 'f' */ {7, 0,0,0,124,128,0,0,0,0,         16,16,16,126,16,144,144,0,0},
X/* 'g' */ {8, 24,36,1,1,1,1,0,62,0,          28,0,34,34,34,34,35,62,0},
X/* 'h' */ {8, 252,0,0,0,0,0,32,28,0,         254,32,32,32,32,32,0,30,0},
X/* 'i' */ {5, 0,0,60,0,0,0,0,0,0,            34,34,190,2,2,0,0,0,0},
X/* 'j' */ {6, 1,1,1,1,0,62,0,0,0,            0,0,0,32,33,190,0,0,0},
X/* 'k' */ {7, 252,0,0,24,0,36,0,0,0,         254,8,8,0,20,0,34,0,0},
X/* 'l' */ {5, 0,0,252,0,0,0,0,0,0,           130,130,254,2,2,0,0,0,0},
X/* 'm' */ {9, 60,32,0,0,60,0,0,32,28,        62,0,32,32,62,32,32,0,30},
X/* 'n' */ {8, 60,0,0,0,0,0,32,28,0,          62,32,32,32,32,32,0,30,0},
X/* 'o' */ {9, 24,36,0,0,0,0,0,36,24,         28,0,34,34,34,34,34,0,28},
X/* 'p' */ {8, 63,0,0,0,0,0,36,24,0,          63,34,34,34,34,34,0,28,0},
X/* 'q' */ {8, 24,36,0,0,0,0,0,63,0,          28,0,34,34,34,34,34,63,0},
X/* 'r' */ {8, 60,0,32,0,0,0,0,0,0,           62,16,0,32,32,32,32,32,0},
X/* 's' */ {9, 0,48,0,0,0,0,0,12,0,           16,2,42,42,42,42,42,32,4},
X/* 't' */ {8, 0,0,248,4,0,0,0,0,0,           32,32,252,32,34,34,34,2,0},
X/* 'u' */ {9, 56,4,0,0,0,0,0,60,0,           60,0,2,2,2,2,2,60,2},
X/* 'v' */ {9, 32,16,8,4,0,4,8,16,32,         32,16,8,4,2,4,8,16,32},
X/* 'w' */ {9, 60,0,4,8,16,8,4,0,60,          60,2,4,8,16,8,4,2,60},
X/* 'x' */ {7, 0,36,24,0,24,36,0,0,0,         34,20,0,8,0,20,34,0,0},
X/* 'y' */ {9, 32,16,8,5,2,4,8,16,32,         32,16,8,5,2,4,8,16,32},
```

```
X/* 'z' */ {8, 4,0,8,0,16,0,32,0,0,           34,38,34,42,34,50,34,34,0},
X/* '{' */ {6, 0,48,72,132,0,0,0,0,0,         16,0,108,0,130,130,0,0,0},
X/* '|' */ {1, 204,0,0,0,0,0,0,0,0,           238,0,0,0,0,0,0,0,0},
X/* '}' */ {6, 0,0,132,72,48,0,0,0,0,         130,130,0,108,0,16,0,0,0},
X/* '~' */ {9, 0,128,0,128,0,64,0,64,0,       64,0,128,0,64,0,32,0,64},
X/*'DEL'*/ {9, 120,132,48,72,72,72,0,132,120, 124,0,0,0,0,0,0,0,0}
X};
+ END-OF-FILE nlqpsdat.c
chmod 'u=rw,g=r,o=' 'nlqpsdat.c'
echo 'SENT: -rw-r-----  1 leo          7848 Jun 24 12:29 nlqpsdat.c'
echo -n 'RCVD: '
/bin/ls -l nlqpsdat.c
exit 0


From -6258999190792946438
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,a1be308beacd6d88,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-07-06 07:46:47 PST
Path: utzoo!attcan!uunet!lll-winken!lll-tis!helios.ee.lbl.gov!pasteur!agate!
saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i061: varlint -- check C code for basic errors
Keywords: C, shar
Message-ID: <4038@saturn.ucsc.edu>
Date: 6 Jul 88 14:46:47 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 522
Approved: koreth@ssyx.ucsc.edu

Submitted-by: decwrl!pacbell!dssch1!dsuthers
Posting-number: Volume 1, Issue 61
Archive-name: varlint

The following is a utility for those of us who program in 'C', but
are lousey typists.  The program checks for:
scanf   sccanf   fscanf  printf   sprintf   fprintf
function calls, and checks their syntax.  It is a SUPPLEMENT to lint
(or a good syntax checking compiler); It is not a replacement for lint.

[Executable is being posted to the binaries group. -sg]

#--------------------------Cut Here--------------------------
#! /bin/sh
# This is a shell archive.  Remove anything before the "#! /bin/sh" line,
# then unpack it by saving it in a file and typing "sh file."
#
# Wrapped by Steven Grimm (koreth) at ssyx on Wed Jul  6 07:38:20 1988
#
# unpacks with default permissions
#
```

```
# Contents : VARLINT.DOC VARLINT.C
#
if `test ! -s VARLINT.DOC`
then
echo "x - VARLINT.DOC"
cat > VARLINT.DOC << '@\Rogue\Monster\'
VARLINT(Local)              'C' Utility       VARLINT(Local)

SYNOPSIS
   varlint <program name> ... <program name>

HISTORY

   This program was developed under Lattice C on an Atari 520 ST. It
   is compatable with (and tested on) DEC 11/70 and AT&T 3B5 processors,
   running SYS V, and earlier versions too.  There is no reason that it
   won't compile on any system.

USAGE

   This program is designed to assist when coding in C.  It double checks
   your syntax for those funtions that have a variable number of arguments,
   and use the same syntax as printf / scanf.

WHEN TO USE IT

   If your program passes lint, and it compiles ok, but crashes any-way,
   or doesn't quite work right, try varlint.

   Mistakes with scanf or printf are one of the most frequent causes of
   intermittant crashes.  "Lint" does not check the syntax, and neither
   do most of the compilers.

CAPABILITIES:

   This program will detect and print out the following conditions.

     1)   More arguments following the control string than there are
          percent signs in the control string.
     2)   Too many arguments as compared to the control string.
     3)   Mis-matches between the argument type and the item in
          the control string.
     4)   Null control string.
     5)   Unknown specifiers after a percent sign in a control string.
        6)   Missing ampersands (&) in front of numbers. (scan functions)
     7)   Will not complain about valid uses. (allows funtion calls as
          arguments, strings as *chars, etc.

RESTRICTIONS:

     1)   It does not yet check to ensure that the variables you name
          are indeed the type you claim.
     2)   The use of a character array is allowed by the scanf / printf
```

```
          functions, but is not yet verified by the program.  You
          are notified of "un-recognized contstruct" in these cases.

LICENSE:

        Copyright 1988 by Daniel B. Suthers, Concord, California, USA.
        License is granted to distribute this program at will, as long as
        the source is also provided free of charge to anyone who
        receives the binaries, or any one who wants it,
        AND all copyright notices are left intact.


@\Rogue\Monster\
else
  echo "shar: Will not over write VARLINT.DOC"
fi
if `test ! -s VARLINT.C`
then
echo "x - VARLINT.C"
cat > VARLINT.C << '@\Rogue\Monster\'
/*##############################################################################
 *
 *   This program is used to find those pesty errors in varags type
 *   functions such as printf.  It only handles those functions defined
 *   in the structure"fu".
 *
 *      By Daniel B. Suthers Mar 3, 1988
 *
 *      Copyright 1988 by Daniel B. Suthers, Concord, California, USA.
 *      License is granted to distribute this program at will, as long as
 *      the source is also provided free of charge to anyone who recieves
 *      the binaries, or to anyone who wants it,
 *   AND all copyright notices are left intact.
 *   The
 *
###move $This_File 755 uprog none /uprog/bin/src/varlint.c
##move $This_File 755 uprog none /usr/uprog/src/bin/varlint.c
 *##############################################################################*/
#include <stdio.h>
#include <ctype.h>

#define VER "2.7"
#define MAX 512
#define NUMNAMES 6

#ifdef pdp11
char APPL[] ="\n##move $This_File 755 uprog none /usr/uprog/bin/varlint\n";
#else
char APPL[]="\n###move $This_File 755 uprog none /usr/lbin/varlint\n";
FILE *fopen();
void scandoos();
#endif
```

```
struct funcdat{
     char nam[9];               /* stores function names*/
     int len;               /* number of chars in names*/
     int pos;               /* position of control string*/
     int scan;
     };

struct funcdat fu[ NUMNAMES] = {
     { "printf", 6, 0, 0},
     { "scanf",  5, 0, 1},
     { "sprintf", 7,1, 0},
     { "sscanf", 6, 1, 1},
     { "fprintf", 7,1, 0},
     { "fscanf", 6, 1, 1}
     };

char *fgets(), *findname();
int in_comment = 0;                  /* flag = 1 when in a comment*/

main(argc, argv, env)
int argc;
char *argv[], *env[];
{
  extern int optind;
  extern char *optarg;
  register int inta;
  char filename[128];
  FILE *infile ;

  while( (inta = getopt( argc, argv, "vVhH")) != EOF)
  {
    switch (inta)
    {
      case 'V':
      case 'v':
     fprintf (stderr, "%s\n"
     , VER);
     return(0);
      case 'h':
      case 'H':
     help();
     return(0);
      default:
     printf("Bad argument\nTry -h for help.\n");
     return(1);
    }
  }
  while( optind < argc )                        /* these would be files*/
  {
    strcpy(filename, argv[optind++]);
    if( (infile = fopen(filename, "r") ) == NULL)
    {
      fprintf(stderr, "Unable to open source file %s\n", filename);
```

```
    }
    else
    {
      printf("Working on %s\n", filename);
      scandoos(infile);
      if( infile != (FILE *) NULL)
     fclose(infile);
    }
  }
return(0);
}                           /*end of main*/

help()
{
  printf(
    "\n\tCopyright 1988 by Daniel B. Suthers, Concord, California, USA.\n");
  printf(
    "\tLicense is granted to distribute this program at will, as long as\n");
  printf(
    "\tthe source is also provided free of charge to anyone who recieves\n");
  printf("\tthe binaries, or to anyone who wants it,\n");
  printf("\tAND all copyright notices are left intact.\n");
  printf("\n\tThis program checks for mismatched arguments in functions\n");
  printf("\tthat accept variable arguments ala printf().\n");
  printf("\tCurrently supports: printf, scanf and their cousins.\n");
  printf("\t(Checks only for # of args and & for integers)\n");
  printf("\n\tThe arguments accepted are:\n\n");
  printf("\t-v\tPrint the version number and exit.\n");
  printf("\t-help\tPrint the most current help.\n");
return(0);
}

#ifdef u3b5
void
#endif
scandoos(infile)
FILE *infile;
{
  register int linenum, reterr, oldline;
  char buff[MAX*3], *buffp;          /* allow 10 lines */
  char tempstr[MAX*3];
  int funcindex;

  linenum = oldline = 0;
  reterr = 0;
  while( fgets(buff, MAX, infile) != (char *)NULL )
  {
    if(reterr == 100 )               /* multi line statement */
    {
      if( oldline == 0)
        oldline = linenum;
      strcat(tempstr,buff);
      strcpy(buff, tempstr);
```

```
  reterr = 0;
}
linenum++;
if( (buffp = findname(buff,&funcindex)) == NULL )
  continue;
if( in_comment )
  continue;
/*
printf("%4d: %s",linenum, buffp);
*/
if( (reterr = parsedo(buffp,funcindex)) > 0)
{
  if( reterr < 100 )
  {
 if( oldline != 0)
 {
   printf("***** Error at line %5d\n%s", oldline, buffp);
   oldline = 0;
 }
 else
   printf("***** Error at line %5d\n%s", linenum, buffp);
  }
  switch(reterr)
  {
 case 1:
   printf("More \"%%\" in control string than there are variables.\n\n");
   break;
 case 2:
   printf("A varible type did not match the \"%%\".\n\n");
   break;
 case 3:
   printf("An undefined variable used.\n\n");
   break;
 case 4:
   printf("An un-recognized construct, might be ok.\n\n");
   break;
 case 5:
   printf("Fewer \"%%\" in control string than there are variables.\n\n");
   break;
 case 6:
   printf("Integer type requires an ampersand (\&).\n\n");
   break;
 case 7:
   printf("Print found a string, expecting an int.\n\n");
   break;
 case 8:
   printf("Unknown %% specifier.\n\n");
   break;
 case 9:
   printf("Re declaration of the function. You sure this is ok?\n\n");
   break;
 case 10:
   printf("Null control string. You sure this is ok?\n\n");
```

```
      break;
     case 100:
      strcpy( tempstr, buff);
      break;
     }
    }                       /* end of if(parsedo) */
    else
      oldline = 0;
  }
}                           /* end of scandoos() */

char *
findname(string, funcind)
char string[];
int *funcind;
{
  char *ptr;
  register int a;

  ptr = string;
  while( *ptr != NULL )
  {
    if( *ptr == '/')                /* set flag if this */
      if( *(ptr + 1) == '*')             /* is start of comment*/
      {
     in_comment = 1;
     ptr += 2;
     continue;
      }
    if( *ptr == '*' )               /* un-set flag if end */
      if( *(ptr+1) == '/')               /* of comment         */
      {
     in_comment = 0;
     ptr +=2;
     continue;
      }
    for( a = 0; a < NUMNAMES; a++)
    {
      if( strncmp( ptr, fu[a].nam, fu[a].len ) == 0 )
      {
     *funcind = a;
     return(ptr);
      }
    }
    ptr++;
  }
return(NULL);
}

int
parsedo(buff, fi)
char buff[];                    /* function string */
int fi;                         /* fu.[] index */
```

```
{
  register int parens, pcntnum, c;
  char *buffp;
  int end=0, inquotes = 0;
  int vartype[124];                   /* for var type */

  parens = 0;
  buffp = buff;
  buffp += fu[fi].len;                /* get past function */
  while( isspace(*buffp) )            /* skip white space  */
    buffp++;
  if( *buffp == NULL)
    return(100);
  if( *buffp != '(' )                 /* not a function */
    return(0);
  if( *(buffp +1) == ')' )            /* function definition */
    return(9);
  buffp++;
  while( isspace(*buffp) )            /* skip white space  */
    buffp++;
  if( *buffp == NULL)
    return(100);
  for(c = 0;c < fu[fi].pos; c++)      /* skip to control string*/
  {                                  /* (stop on comma) */
    while( (*buffp != NULL) && (*buffp != ',') )
     buffp++;
    if( *buffp == NULL)
      return(100);
    buffp++;
  }
  if( *buffp == NULL)
    return(100);
  while( isspace( *(buffp)) )         /* for ' , "test"'   */
    buffp++;
  if( *buffp == NULL)
    return(100);
  if( *buffp != '"')                  /* should be on first quote*/
    return(4);
  if( *(buffp+1) == '"')              /* dont want "" as cntl str */
    return(10);
                            /* count the percent signs */
                            /* starting on first quote */
  pcntnum = 0;
  end = 0;
  while( *buffp++ != NULL )
  {
    switch(*buffp)
    {
     case '"':
       if( *(buffp-1) != '\\')        /*    \" is not end   */
         end = 1;
       else if( *(buffp-2) == '\\')       /*    \\"  is end     */
         end = 1;
```

```
  break;
case '%':
  if( *(buffp-1) != '%')
    pcntnum++;
  else
  {
    pcntnum--;                  /* delete first % of pair */
    continue;
  }
  if( *(buffp+1) == '*' )
  {
    if(  fu[fi].scan == 1)
    {
      pcntnum--;            /* no matching variable */
    }
    else
    {
      pcntnum++;
    }
  }
                        /*##########################
                        # load array with var type
                        ##########################*/
                        /* throw away justification*/
  if( *(buffp +1 ) == '-' || *(buffp +1) == '+')
    buffp++;
                        /* remove blanks and sharps */
     while( isspace( *(buffp + 1)) || *(buffp + 1) == '#' )
       buffp++;
                        /* throw away size spec */
  while( isdigit(*(buffp+1)) || *(buffp+1) == '.' )
    buffp++;
  switch (*(buffp + 1) )        /* look at next char for type */
  {
    case 's':
    case '[':
    case 'c':
      vartype[pcntnum] = 's';
      break;
    case 'd':
    case 'l':
    case 'h':
    case 'i':
    case 'o':
    case 'x':
    case 'u':
    case 'f':
    case 'e':
    case 'g':
      vartype[pcntnum] = 'i';
      break;
    case '%':               /* no action */
    case '*':
```

```
        break;
      default:
        printf("error at %s\n\n", buffp + 1);
        return(8);
        break;
    }
    break;
  }
  if( end )
   break;
}
parens = 0;
inquotes = 1;                       /* start on last " */
c = 1;
while( *buffp != NULL )             /* look for variables */
{
  switch( *buffp )
  {
    case '(':
     parens--;
     break;
   case ')':
     parens++;
     break;
   case '"':               /* for strings */
     if( parens == 0)              /* at the base level */
     {
       if( *(buffp-1) != '\\' )         /*    \" is not end of str */
       {
         inquotes = !inquotes;
       }
       else if( *(buffp-2) == '\\')     /*    \\"  is end     */
         inquotes = !inquotes;
     }
     break;
   case ',':
     if( parens == 0 && inquotes != 1)  /* commas outside of func*/
     {
       pcntnum --;
       buffp++;
       while( isspace(*buffp) )
         buffp++;
          if( *buffp == NULL)
            return(100);
       if( fu[fi].scan == 1 )
       {
         if( vartype[c] == 'i' && *buffp != '\&' )
        return(6);
       }
       else if( vartype[c] == 'i' && *buffp == '"')
           return(7);
     c++;
     continue;                 /* buffp already ++ */
```

```
      }
      break;
    }
    if( parens == 1)                /* hit end of function */
    {
     if( pcntnum > 0 )
       return(1);
     if( pcntnum < 0)
       return(5);
     return(0);
    }
    buffp++;
  }
  if( parens < 1 )
    return(100);
return(0);
}                           /* end of parsedo() */


@\Rogue\Monster\
else
  echo "shar: Will not over write VARLINT.C"
fi
# to concatenate archives, remove anything after this line
exit 0


From -1847234608390834686
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,9b05e3af3f7f538e,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-07-15 09:41:38 PST
Path: utzoo!attcan!uunet!lll-winken!lll-lcc!ames!mailrus!uwmcsd1!ig!agate!saturn!
ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i062: flip27 -- double-sided disk copier  part01/02
Keywords: Megamax
Message-ID: <4165@saturn.ucsc.edu>
Date: 15 Jul 88 16:41:38 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1019
Approved: koreth@ssyx.ucsc.edu

Submitted-by: uunet!mcvax!mirsa.inria.fr!colas (Colas Nahaboo)
Posting-number: Volume 1, Issue 62
Archive-name: flip27/part01

#--------------------------Cut Here--------------------------
#! /bin/sh
# This is a shell archive.  Remove anything before the "#! /bin/sh" line,
# then unpack it by saving it in a file and typing "sh file."
#
```

```
# Wrapped by Steven Grimm (koreth) at ssyx on Fri Jul 15 09:32:22 1988
#
# unpacks with default permissions
#
# Contents : FLIP.C
#
if `test ! -s FLIP.C`
then
echo "x - FLIP.C"
cat > FLIP.C << '@\Rogue\Monster\'
/*****************************************\
*                           *
* flip: Unprotected disk copier           *
* By Colas NAHABOO (INRIA sophia-antipolis)  *
*                           *
* Compile with MEGAMAX C v1.1 (in-line asm)  *
*                           *
\*****************************************/

#define VERSION "2.7"
#define DATE  "Jun 25 88"

char           *header =
"\33E\33e\33v\033p\
FLIP: disk copier by Colas Nahaboo     v%-5s %-9s           Free Ram:%4ldK\
\033q\n\n";

#include <osbind.h>
#include <stdio.h>

#define BLANK (0xE5E5)      /* std sector format value */
#define MAGIC (0x87654321L) /* magic number value */
unsigned char *ROMS_YEAR = 0x00fc0018L + 3L; /* year of the roms */
#define MEMORY_LEFT 0       /* bytes not eaten by main buffer */
#define TRACK_RETRY 4       /* number of full track retrys */
#define NO_DISK        0    /* no disk in drive */
#define SOURCE_DISK    1    /* source disk in drive */
#define DEST_DISK  2    /* destination disk in drive */

                  /* list of fatal errors */
#define NORMAL          10
#define USER_ABORT 1
#define SYNTAX_ERROR    2
#define NO_DISK_IN_MEMORY 3
#define NO_OVERRIDE     4
#define NO_DRIVE_B 5
#define DOUBLE_ON_BACK  6
#define NO_FLIP_ON_DOUBLE 7
#define BAD_OPTION 8
#define NO_MEMORY  9

int bad_option     =    ' ';
```

```
/* twister interface */
extern          twister();  /* adress of formatter */
char            buffer_data[0x3FFF]; /* 16k of buffer */
extern long     buffer;     /* format track buffer */
extern int      thedisk,    /* A=0, B=1 */
                dblsided,   /* SS=0, DS= -1 */
                sectoroffset,    /* 0 = sectors 0 to 10, 1= 11 to 20 */
                flipped,    /* normal=0, back side=1 */
         keypress,/* 1 if key aborted format */
                starttrack, /* 0 or more */
                endtrack;   /* 80 or more */
extern int      badflag;    /* return code */
extern int    new_roms;/* are we running with the new roms? */

/* flip's globals */
long            buffer_size = 0L;     /* max size of buffer */
jmp_buf         start_dest; /* insert dest */
int      drives_content[2];/* what is there in the drives ? */

int               /* disk parameters read from boot sect */
                drive,      /* drive: 0=A, 1=B */
                sides,      /* 0=SS, 1=DS */
                tracks,     /* 80 normally */
                st,         /* sectors per track (9 or 10) */
                bps;        /* bytes per sector (512) */

int               /* options */
         verify_source,     /* just verify the source */
         read_source,  /* read source */
         zero_disk,    /* prepares blank image in memory */
         write_dest,   /* write dest */
                format,     /* do format */
                Verify,     /* do verify formatting */
                verify,     /* do verify writing */
         override,/* do not read boot sector */
         single,       /* Single Sided */
         nine_sectors, /* 9 sect/track */
                interactive,     /* interactive */
         play_it_again_sam, /* user hit "return" */
                use_track_map;   /* to use built-in track map */

int             d1=0, d2=0, s1=0, s2=0; /* drive/side source/dest */

#define TRACK_MAP_SIZE  99
char            track_map[TRACK_MAP_SIZE + 1];
char            full_track_map[TRACK_MAP_SIZE + 1]; /* for multipasses */

int             one_valid_copy_done = 0;  /* used to allow "r" */
int             multipass;  /* when not enough mem */
int      first_pass,
         last_pass;

jmp_buf         start;
```

```
char           *buf = NULL;
char           *malloc(), *index(), *get_drive();
unsigned is_non_executable();
extern char    *help_text[];
int             errno;

char            rep[40], default_rep[40]; /* input buffer */

/*
 * parse the command line and set options
 * return 1 if error, 0 if OK
 */

int
parse_command()
{
     printf("Command (? for help) (CR= %s ): ", default_rep);
     fflush(stdout);
     gets(rep);
     /* parse main options */
     verify_source = read_source = zero_disk = write_dest = format =
     verify = Verify = use_track_map = override = nine_sectors = single =
     interactive = d1 = d2 = s1 = s2 = play_it_again_sam = 0;
     if (!rep[0]) {          /* void rep ==> default */
         strcpy(rep, default_rep);
         play_it_again_sam = 1;
     }
     switch(rep[0]){
     case '?':     /* need help */
         more(help_text);
         longjmp(start, NORMAL);
     case 'q':     /* quit */
         exit(0);
     case 'v':
         verify_source = 1;
         get_options(get_drive(SOURCE_DISK, rep+1, 1), "os9ti");
         return 0;
     case 'r':
         if (!one_valid_copy_done)
          longjmp(start, NO_DISK_IN_MEMORY);
         get_options(get_drive(DEST_DISK, rep+1,1), "fvVi");
         write_dest = 1;
         return 0;
     case 'f':
         override = 1;
         get_options(get_drive(DEST_DISK, rep+1,1), "s9tvVfi");
         zero_disk = 1;
         write_dest = 1;
         return 0;
     case 'a': case 'b':
         get_options(
          get_drive(DEST_DISK,
                get_drive(SOURCE_DISK, rep, 0),
```

```
                1),
             "os9tvVfi");
            read_source = write_dest = 1;
            return 0;
        default :
            longjmp(start, SYNTAX_ERROR);
        }
}

char *
get_drive(drive, string, is_optional)
int drive;
char * string;
int is_optional;
{
    char *ptr =string;
    int side = 0, default_drive = 0;

    if(!index("abAB", string[0]))
     if(is_optional)
         default_drive = 1;      /* drive A is default */
     else
         longjmp(start, SYNTAX_ERROR);
    if(!default_drive && (index("12", string[1]))) {
     ptr ++;
     side = (toupper(*ptr) == '2' ? 1 : 0);
    }
    if(drive == DEST_DISK){
     d2 = (default_drive ? 0 : (toupper(*string) == 'B' ? 1 : 0));
     s2 = side;
    }else{
     d1 = (default_drive ? 0 : (toupper(*string) == 'B' ? 1 : 0));
     s1 = side;
    }
    return (default_drive ? string : ptr +1);
}

get_options(string, options)
char *string, *options;
{
    while(*string){
     if(!index(options, *string)) {
         bad_option = *string;
         longjmp(start, BAD_OPTION);
     }
     switch(*string){
     case 'o': override = 1; break;
     case 's': single = 1; break;
     case '9': nine_sectors = 1; break;
     case 't': use_track_map = 1; break;
     case 'v': verify = 1; break;
     case 'V': Verify = 1; break;
     case 'f': format = 1; break;
```

```
     case 'i': interactive = 1; break;
     }
    string++;
    }
    if(!override && (single || nine_sector))
     longjmp(start, NO_OVERRIDE);
}

main()
{

    buffer = (long) buffer_data;/* init twister */
    sectoroffset = 0;       /* mem alloc */
    buffer_size = (long) Malloc(-1L) - MEMORY_LEFT;
    buf = (char *) Malloc(buffer_size);
    if(((unsigned) *ROMS_YEAR) > (unsigned) 0x86)
     new_roms = 1;
    else
     new_roms = 0;
    if(is_drive_B())        /* default command */
     strcpy(default_rep, "bfv");
    else
     strcpy(default_rep, "afv");
    printf(header, VERSION, DATE, buffer_size / 1024);  /* resets screen */
    switch(setjmp(start)){
    case 0:case NORMAL:          /* initial */
     break;
    case USER_ABORT:             /* user abort */
     printf("\007Ok, aborted...\n");
     break;
    case SYNTAX_ERROR:           /* syntax error */
     printf("\007SYNTAX ERROR!\n");
     break;
    case NO_DISK_IN_MEMORY:
     printf("\007No disk in memory!\n");
     break;
    case NO_OVERRIDE:
     printf("\007Options d and 9 requires option o\n");
     break;
    case NO_DRIVE_B:
     printf("\007Sorry, no drive B present...\n");
     break;
    case DOUBLE_ON_BACK:
     printf("\007A double sided disk on back? use override!\n");
     break;
    case NO_FLIP_ON_DOUBLE:
     printf("\007You cannot flip double-sided disks!\n");
     break;
    case BAD_OPTION:
     printf("\007bad option: %c\n", bad_option);
     break;
    case NO_MEMORY:
     printf("\007Sorry, not enough memory...\n");
```

```
     break;
    }
    while (1) {
     /* we dont know what's in the drives */
     drives_content[0] = drives_content[1] = NO_DISK;
     /* prompts and parse command and options */
     parse_command();
     /* check for drive B */
     if ((d1 || d2) && (!is_drive_B()))
         longjmp(start, NO_DRIVE_B);
     /* prompts for track_map */
     if (use_track_map)
         input_track_map();
     /* examine source disk */
     if(read_source || verify_source || zero_disk)
         examine_disk(d1, s1);
     /* check run time consistency */
     if (sides == 2) {
         if (s1 == 1)
             longjmp(start, DOUBLE_ON_BACK);
         if (s1 != s2)
          longjmp(start, NO_FLIP_ON_DOUBLE);
     }
     strcpy(default_rep, rep);   /* seems ok then take it as default */

     /* copy disk */
     if((d1 != d2) &&   /* suppose user had time to put dest */
          (read_source || verify_source))
         drives_content[d2] = DEST_DISK;
     do_copy_disk();
     printf("\007OK, done.\n");
    }
}

/*
 * do_copy_disk:
 * do a one-pass or multipass copy
 */

do_copy_disk()
{
     int             track_number = 0, i, j, to_do;

     multipass = 0;
     if (read_source) {
         for (i = 0; i < TRACK_MAP_SIZE; i++)
          track_number += (int) track_map[i];
         if(buffer_size < (sides * bps * st * (long) track_number))
             multipass = 1;
         one_valid_copy_done = !multipass; /* let previous value of
               multipass inchanged if no read_source */
     }
     if(zero_disk) one_valid_copy_done = 0;
```

```
    bcopy(track_map, full_track_map, TRACK_MAP_SIZE);
    if(!multipass){
        copy_disk();
    }else{
        for(j=0; j<TRACK_MAP_SIZE; j++)
         track_map[j] = '\0';
        track_number = buffer_size / (sides*bps*st);
        i = to_do = 0;
        first_pass = 1;
        last_pass  = 0;
        while(i < TRACK_MAP_SIZE){
         if(full_track_map[i]){
             to_do++;
             track_map[i] = '\001';
         }
         if(to_do == track_number){
             copy_disk();
             to_do = first_pass = 0;
             for(j=0; j<TRACK_MAP_SIZE; j++)
              track_map[j] = '\0';
         }
         i++;
        }
        first_pass = 0;
        last_pass = 1;
        copy_disk();
    }
}


/*
 * examine_disk: reads boot sector to determine type of disk.
 */

/* describe the BOOT sector by offsets in the boot sector */
#define b_bootbranch 0
#define b_oem 0x02
#define b_serial 0x08
#define b_bps 0x0B
#define b_spc 0x0D
#define b_res 0x0E
#define b_nfats 0x10
#define b_ndirs 0x11
#define b_nsects 0x13
#define b_media 0x15
#define b_spf 0x16
#define b_spt 0x18
#define b_nsides 0x1A
#define b_nhide 0x1C
#define b_boot 0x1E
#define b_checksum 0x01FE

/* 2 chars ==> 1 int */
```

```
#define canshort(x,y) (((unsigned)(x))|(((unsigned)(y))<<8))

unsigned char   bb[512];

randomize_boot()
{
    long serial;
    unsigned non_exec = is_non_executable(buf);

    if((!multipass || first_pass) && track_map[0]) {
     serial = Random();
     buf[b_serial] = (char) ((serial >> 16) & 255L);
     buf[b_serial+1] = (char) ((serial >> 8) & 255L);
     buf[b_serial+2] = (char) (serial & 255L);
     update_checksum(buf, non_exec);
    }
}

/*
 * prepares a blank image:
 * boot sector, & zero tracks 0&1 (if enough memory)
 */

do_zero_disk()
{
    unsigned nsects;

    if(buffer_size < (sides * bps * st * 2L)) longjmp(start, NO_MEMORY);
    bfill(buf, '\0', (long) (sides * bps * st * 2L));
    Protobt(buf, 0x02000000L, ((sides == 2) ? 3 : 2), 0);
    buf[b_spt] = (char) (st & 0xFF);
    nsects = (unsigned) (sides * st * (1 + max_track()));
    buf[b_nsects] = (char) (nsects & 255);
    buf[b_nsects+1] = (char) ((nsects >> 8) & 255);
    update_checksum(buf, (unsigned) 1);
    printf("Formatting %s sided disk, %d tracks and %d sectors.\n",
     ((sides == 1) ? "single" : "double"), 1+max_track(), st);
}

int
max_track()
{
    int i;

    for (i= TRACK_MAP_SIZE -1; i>=0; i--)
     if(full_track_map[i]) return i;
    return 0;
}

examine_disk(drive, side)
int             drive, side;
{
    int             i;
```

```
    if (override) {
     sides = (single ? 1 : 2);
     bps = 512;
     st = (nine_sectors ? 9 : 10);
     tracks = 80;
    } else {
     if (drives_content[drive] != SOURCE_DISK) {
         prompt_for_disk(SOURCE_DISK, drive, side);
     }
     while (Floprd(&bb, 0L, drive, 1, 0, side, 1)) {
         printf("\007Cannot read boot block! CR/SP=retry, other=abort: ");
         fflush(stdout);
         if (!index("\r\n ", (char) Cnecin()))
          longjmp(start, USER_ABORT);
     }
     if(!is_non_executable(bb)) {
    printf("\007EXECUTABLE BOOT! at $3A, %2x %2x %2x -- check for virus\n",
         (int) bb[58], (int) bb[59], (int) bb[60]);
     }
     printf("Disk in drive %c:", 'A' + drive);
     printf(" %d side", sides = (int) bb[b_nsides]);
     printf("%s, %d tracks,", (sides == 2 ? "s" : ""),
            tracks = ((canshort(bb[b_nsects], bb[b_nsects + 1]) /
                (int) bb[b_spt]) /
               (int) bb[b_nsides]));
     printf(" %d sectors of", st = (int) bb[b_spt]);
     printf(" %d bytes\n", bps =
            canshort(bb[b_bps], bb[b_bps + 1]));
     if ((bps != 512) || ((st != 10) && (st != 9)) || (tracks != 80)) {
         printf("\007WARNING -- this is not a normal disk!\n");
     }
     if (sides < 1 || sides > 2 ||
         bps != 512 ||
         st > 11 || st < 9 ||
         tracks < 1 || tracks > TRACK_MAP_SIZE) {
         printf("***** Strange disk! ... copying normally ");
         printf("(1 side, 80 tracks of 9 sectors)\n");
         override = 1;
         sides = 1;
         bps = 512;
         st = 9;
         tracks = 80;
     }
    }
    if (!use_track_map) {
     for (i = 0; i < tracks; i++)
         track_map[i] = '\001';
     for (i = tracks; i < TRACK_MAP_SIZE; i++)
         track_map[i] = '\0';
    }
}
```

```
/*
 * format disk using TWISTER
 * returns 0 if OK, aborts and returns 1 on error.
 */

int
format_disk(f_drive, f_double, f_side)
int             f_drive, f_double, f_side;
{
    int             i;
    char     *map;

    if(multipass)
        map = full_track_map;
    else
     map = track_map;
    thedisk = f_drive;
    dblsided = (f_double ? -1 : 0);
    flipped = f_side;
redo:
    i = 0;
    while (i < TRACK_MAP_SIZE) {
     if (map[i]) {
         starttrack = i;
         while (map[i++]);
         endtrack = i - 1;
         keypress = 0;
         Supexec(twister);
         if(keypress) {
          longjmp(start, USER_ABORT);
         }
         if (badflag){
          printf("\n\007FORMAT Error! CR/SP = retry whole formatting,");
          printf(" other=abort\n");
          printf("(disk might be WRITE_PROTECTED?)\n");
          if (!index("\r\n ", (char) Cnecin()))
              longjmp(start, USER_ABORT);
          else
              goto redo;
         }
     } else {
         i++;
     }
    }
}

/*
 * copy_disk: where the copy is really done
 */

copy_disk()
{
    if((read_source || verify_source)
```

```
        && (drives_content[d1] != SOURCE_DISK))
        prompt_for_disk(SOURCE_DISK, d1, s1);
    if (read_source) {      /* source */
     read_disk(d1, s1); /* read */
     next_line();
    }
    if (verify_source && (!multipass || first_pass)) {
     verify_disk(d1, s1);
     next_line();
    }

    setjmp(start_dest);     /* destination */
    if(zero_disk) do_zero_disk();              /* blank image */
    if(!override) randomize_boot();            /* change serial */
    if((format || Verify || write_dest || verify))
     prompt_for_disk(DEST_DISK, d2, s2);
    if (format && (!multipass || first_pass)) {         /* format */
     format_disk(d2, sides - 1, s2);
     next_line();
    }
    if (Verify && (!multipass || first_pass)) {         /* Verify */
     verify_disk(d2, s2);
     next_line();
    }
    if (write_dest) {       /* write */
     write_disk(d2, s2);
     next_line();
    }
    if (verify && (!multipass || last_pass)) {     /* verify */
     verify_disk(d2, s2);
     next_line();
    }
}

/*
 * return 1 if skip, 0 if no key, longjmps to start if abort
 */

abort_on_key(track,side)
int             track,side;
{
    if (Cconis()) {
     if ((int) Cnecin() == 0x1b)
         longjmp(start, USER_ABORT);
     printf("\n\007KEY PRESSED! CR/SP = continue, ");
     printf("i=interactive, %s other abort:\n",
            (track == -1 ? "" : " S = skip,"));
     switch ((int) Cnecin()) {
     case ' ':
     case '\n':
     case '\r':
         return 0;
     case 'i':
```

```
     case 'I':
         interactive = 1;
         return 0;
     case 's':
     case 'S':
         if (track != -1) {
          printf("Ok, skipping track %d, side %d (sects #%d - #%d)\n",
                 track + 1, side +1,
                 ((track * sides) + side)* st,
                 ((track * sides) + side + 1) * st);
          return 1;
         }
     }
     longjmp(start, USER_ABORT);
    }
}

int
retry(s, t, side, restart)
char           *s;
int             t, side;
long           *restart;
{
    printf("\n\007** %s ERROR track %d ,side %d (s. #%d - #%d) ",
        s, t + 1, side + 1,
        (t * sides + side) * st,
        (t * sides + side + 1) * st);
    printf("CR/SP retry, S skip, %sother abort",
        (restart ? "BS=OTHER disk " : ""));
    fflush(stdout);
    switch ((int) Cnecin()) {
    case ' ':
    case '\r':
    case '\n':
     return 1;
    case 's':
    case 'S':
     printf("\n OK, skipping track %d, side %d\n", t + 1, side + 1);
     return 0;
    case '\010':
     if (restart)
         longjmp(restart, 1);
    default:
     longjmp(start, USER_ABORT);
    }
}

/*
 * sector_read
 * reads one track into ptr, reading sector by sector
 * returns: 0=ok, 1=bad, 2= skip.
 */
```

```
int
sector_read(ptr, foo, drive, sect1, track, side, n)
char           *ptr, *foo;
int             drive, sect1, track, side, n;
{
    int             i, result = 0, pass;
    int             pass_ok[4];

    for (pass = 0; pass < 4; pass++) {
     pass_ok[pass] = 1;
     for (i = 0; i < n; i++) {
         if (abort_on_key(track,i))
          return 2;
         switch (one_sector_read(
           (pass ? buffer_data + (pass - 1) * 5120 : ptr) + (i * bps),
                       foo, drive, i + sect1, track, side)) {
         case 1:
          if (!result)
              printf("\nTrack %d, side %d: sectors ",
              track + 1, side + 1);
          result = 1;
          printf("%d (#%d) ", i + 1, (track * sides + side) * st + i);
          fflush(stdout);
          bfill(ptr + i * bps, '\0', (long) bps); /* fills bad sector with */
                              /* 0's */
         case 2:
          pass_ok[pass] = 0;
         }
     }
     if (result) {
         printf("are bad!\n");
         fflush(stdout);
         return result;
     }
    }
    if (pass_ok[0])
     return 0;     /* if any pass is good, use it */
    {
     int             i;

     for (i = 1; i < 4; i++) {
         if (pass_ok[i]) {
          bcopy(buffer_data + 512 * (i - 1), ptr, 512 * n);
          return 0;
         }
     }
    }
    if (check_random(track, side, n, ptr, buffer_data)) {
     return 1;     /* random read */
    } else {
     printf("\nTrack %d, side %d, (sects #%d - #%d) read sect by sect\n"
            ,track + 1, side + 1,
            (track * sides + side) * st,
```

```
            (track * sides + side + 1) * st);
     return 1;     /* no but don't trust! */
    }
}

/*
 * check_random:
 * check tracks buffer to see if equal, return 1 if error
 */

int
check_random(track, side, n, buf0, buf1)
int             track, side, n;
char           *buf0, *buf1;
{
    int             sect, i, bad = 0;
    char           *p0, *p1, *p2, *p3;

    for (sect = 1; sect <= n; sect++) {
     p0 = buf0 + (sect - 1) * 512;
     p1 = buf1 + (sect - 1) * 512;
     p2 = buf1 + (sect - 1) * 512 + 5120;
     p3 = buf1 + (sect - 1) * 512 + 10240;
     for (i = 0; i < 512; i++, p0++, p1++, p2++, p3++) {
         if ((*p0 != *p1) || (*p0 != *p2) || (*p0 != *p3)) {
          if (!bad) {
              printf("\n\007DIFFERENT READS ");
              printf("on track %d, side %d, sects ",
                  track + 1, side + 1);
              bad = 1;
          }
          printf("%d (#%d) ", sect,
          (track * sides + side) * st + sect - 1);
          break;
         }
     }
    }
    if(bad) {
        printf("\n");
     fflush(stdout);
    }
    return bad;
}


/*
 * one_sector_read:
 * reads a sector and the 2 outside it
 * return 0 if ok, 1 if bad, 2 if forced to copy sector alone
 */

char           *private_buffer[512 * 3];
```

```
int
one_sector_read(buf, foo, drive, sect, track, side)
char           *buf, *foo;
int             drive, sect, track, side;
{
    if (Floprd(private_buffer + (sect == 1 ? 512 : 0), foo, drive,
            (sect == 1 ? 1 : sect - 1), track, side,
            ((sect == 1) || (sect == st) ? 2 : 3))) {
     if (Floprd(private_buffer + 512, foo, drive, sect, track, side, 1)){
         return 1;
     } else {
         bcopy(private_buffer + 512, buf, 512);
         return 2;
     }
    }
    bcopy(private_buffer + 512, buf, 512);
    return 0;
}

next_line()
{
    Crawio((int) '\r');
    Crawio((int) '\n');
}

bcopy(from, to, count)
register char  *from, *to;
int             count;
{
    register char  *last = from + count;

    while (from != last)
     *to++ = *from++;
}

bfill(from, octet, count)
register char  *from, octet;
long            count;
{
    register char  *last = from + count;

    while (from != last)
     *from++ = octet;
}

/*
 * is_drive_B
 * return 1 if drive B is there
 */

int
is_drive_B()
{
```

```
    long            usermode = Super(0L);
    int             result = *(int *) 0x4a6;   /* _nflops */

    Super(usermode);
    return (result - 1);
}

/*
 * retry_track
 * retries plain track
 */

int
retry_track(track, side)
int             track, side;
{
    printf("\n\007** Track-read error track %d ,side %d \n", track + 1, side + 1);
    printf("CR/SP retry, ESC aborts, other go to sector_read: ");
    fflush(stdout);
    switch ((char) Cnecin()) {
    case ' ':
    case '\r':
     return 1;
    case 0x1b:
     longjmp(start, USER_ABORT);
    default:
     return 0;
    }
}

verify_disk(drive, side)
int             drive, side;
{
    char           *ptr, *map;
    int             t, n;

    if(multipass)
        map = full_track_map;
    else
     map = track_map;
    ptr = buffer_data;
    for (t = 0; t < TRACK_MAP_SIZE; t++) {
     if (map[t]) {
         for (n = side; n < (side + sides); n++) {
          if (Floprd(ptr, 0L, drive, 1, t, n, st))
              while (retry("Verify", t, n, start_dest) &&
                  Floprd(ptr, 0L, drive, 1, t, n, st));
          abort_on_key(t,n);
         }
         Crawio((int) 'v');
     }
    }
}
```

```
write_disk(drive, side)
int             drive, side;
{
    char           *ptr;
    int             t, n, last_track;

    ptr = buf;
    if(zero_disk){          /* formatting is a special case */
     last_track = 2;
    }else{
     last_track = TRACK_MAP_SIZE;
    }
    for (t = 0; t < last_track; t++) {
     if (track_map[t]) {
         for (n = side; n < (side + sides); n++) {
          if (Flopwr(ptr, 0L, drive, 1, t, n, st))
              while (retry("Write", t, n, start_dest) &&
                  Flopwr(ptr, 0L, drive, 1, t, n, st));
          ptr += (long) bps *st;

          abort_on_key(-1,n);
         }
         Crawio((int) '.');
     }
    }
}

read_disk(drive, side)
int             drive, side;
{
    char           *ptr = buf, c;
    int             t, n;
    int             full_track_retry;

    /* we need at least room for 1 track! */
    if(buffer_size < (sides * bps * st * 1L)) longjmp(start, NO_MEMORY);
    for (t = 0; t < TRACK_MAP_SIZE; t++) {
     if (track_map[t]) {
         c = ',';
         for (n = side; n < (side + sides); n++) {
          if (Floprd(ptr, 0L, drive, 1, t, n, st)) {
              c = ';';
              full_track_retry = 1;
              while(1) {
               abort_on_key(-1,n);
               if (!Floprd(ptr, 0L, drive, 1, t, n, st))
                   goto good_retry;
               full_track_retry++;
               if (full_track_retry >= TRACK_RETRY) {
                   if (interactive && retry_track(t, n)) {
                       full_track_retry = 1;
                   }else{
```

```
                    break;
                }
            }
            }
            /* 4 retries done */
            if (1 == sector_read(ptr, 0L, drive, 1, t, n, st)) {
             if (interactive) {
                 while (retry("Read", t, n, 0L) &&
                 (1 == sector_read(ptr, 0L, drive, 1, t, n, st)));
             }
            }
         }
     good_retry:
         ptr += (long) (bps * st);
         abort_on_key(-1,n);
        }
        Crawio((int) c);
     }
    }
}

input_track_map()
{
    char            text[10];
    int             i, strack, etrack;

    if(play_it_again_sam) {
     bcopy(full_track_map, track_map, TRACK_MAP_SIZE);
     return;
    }
    for (i = 0; i < TRACK_MAP_SIZE; i++)
     track_map[i] = '\0';
    while ((printf("Enter start track (CR = done): "),
         gets(text), (int) text[0])) {
     strack = atoi(text) - 1;
     printf("Enter end track (CR = %d): ", strack + 1);
     gets(text);
     if (!text[0])
         etrack = strack;
     else
         etrack = atoi(text) - 1;
     if ((etrack > TRACK_MAP_SIZE) ||
         (strack > etrack) ||
         (strack < 0)) {
         printf("Sorry, not a valid entry: %d...%d\n",
             strack + 1, etrack + 1);
     } else {
         for (i = strack; i <= etrack; i++)
          track_map[i] = '\001';
         printf("Ok, range %d...%d will be done\n",
             strack + 1, etrack + 1);
     }
    }
```

```
}

/*
 * prompt_for_disk:
 * ask user to insert disk if we think it is not in drive
 */

prompt_for_disk(disk, drive, side)
int disk, drive,side;
{
    if(drives_content[drive] != disk){
     printf("\nInsert %s disk in drive %c (side %d),",
         (disk == SOURCE_DISK ? "SOURCE" : "DESTINATION"),
         (char)(drive + 'A'), side + 1);
     printf("CR/SP = ok, other = abort\n");
     if (!index("\r\n ", (char) Cnecin())){
         if(disk == SOURCE_DISK)
          one_valid_copy_done = 0;
         longjmp(start, USER_ABORT);
     }else{
         drives_content[drive] = disk;
     }
    }
}

/*
 * update_checksum:
 * compute checksum of boot buffer and set it (last int of 256 ints)
 */

update_checksum(block, is_boot_non_executable)
unsigned block[];
unsigned is_boot_non_executable;
{
    int i; unsigned sum = 0;
    for(i=0; i<255; i++)
     sum += block[i];
    block[255] = (unsigned) 0x1234 - sum + is_boot_non_executable;
}

unsigned
is_non_executable(block)
unsigned block[];
{
    int i; unsigned sum = 0;
    for(i=0; i<=255; i++)
     sum += block[i];
    return (sum == (unsigned) 0x1234 ? (unsigned) 0 : (unsigned) 1);
}

@\Rogue\Monster\
else
  echo "shar: Will not over write FLIP.C"
```

```
fi
echo "Finished archive 1 of 2"
# to concatenate archives, remove anything after this line
exit 0


From 3234433694143590139
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,c654c867d702bfa2,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-07-15 09:43:06 PST
Path: utzoo!attcan!uunet!lll-winken!lll-lcc!ames!mailrus!uwmcsd1!ig!agate!saturn!
ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i063: flip27 -- double-sided disk copier  part02/02
Keywords: Megamax
Message-ID: <4166@saturn.ucsc.edu>
Date: 15 Jul 88 16:43:06 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1306
Approved: koreth@ssyx.ucsc.edu

Submitted-by: uunet!mcvax!mirsa.inria.fr!colas (Colas Nahaboo)
Posting-number: Volume 1, Issue 63
Archive-name: flip27/part02

#--------------------------Cut Here-------------------------
#! /bin/sh
# This is a shell archive.  Remove anything before the "#! /bin/sh" line,
# then unpack it by saving it in a file and typing "sh file."
#
# Wrapped by Steven Grimm (koreth) at ssyx on Fri Jul 15 09:32:22 1988
#
# unpacks with default permissions
#
# Contents : MAN.C TWST.C MAKEFILE README
#
if `test ! -s MAN.C`
then
echo "x - MAN.C"
cat > MAN.C << '@\Rogue\Monster\'
/* FLIP HISTORY & MANUAL */

/* HISTORY:
 *=========
 * v 2.7  June 25 88
 *     Twister code works on NEW ROMS!
 *   2.6c when command is just "CR", keep the old track map
 *   2.6b tells when copies an executable boot to detect virii
 *   2.6a bug de 'r' repaired
 *        No checksum done if no previous checksum
 *        option 's' for single instead of double (double=default)
```

```
* v 2.6  Feb 27 88
*   2.5c re-designing of command parsing
*     formats disk ('f' command)
*     randomization of serial # at each copy
*   2.5b t can override disk
*     ESC aborts without confirm
*   2.5a no ESC to another disk, BS!
*     aborts on key even in format
*     man in another C file
* v 2.5  released
*  2.4.5 write-protect detected
*  2.4.3 option v to verify source only
*     multipass copies possible
*  2.4.2 all sects# outputted on any kind of error.
*        twister modified to accept starttrack parameter
*     track_map to know what to copy
*     t option to specify track
*     r takes options fvV
*  2.4.1 i for interactive: default no stop
*     V verify after format (case-sensitive options)
*     bug of verify reading into main buffer repaired
* v 2.4: Jan 15 88
*  2.3.2 If esc on insert abort
*     track_read first
*     no more skip opt
*     # of sect printed
*  2.3.1 Drive B presence is really tested
* v 2.3: Dec 28 87
*  2.2.3 comparisons were not done!!! -- repaired
*     copy by 3 sectors
*        stops if copy sector only by 1
*  2.2.2 bug de a1a1 repare
*  2.2.1 no verify by default
*     r for re-copy
* v 2.2: Dec 19 87
*  2.1.1 prints buffer size on start
*     sector_read done 4 times with comparison!
*     RETURN repeats last command
* v 2.1: Nov 28 87
*  2.0.1 retrys in writes and verify work now.
*     verify there is a drive B before copying
* v 2.0: Nov 22 87
*  1.6.3 verify is another pass
*  1.6.2 format does not verify anymore
*     format outputs a char
*  1.6.1 uses TWISTER's code mmtwst.c to format
* v 1.6: Nov 11 87
*  1.5.4 abort on ESC
*        option "2" for 2-sided.
*  1.5.3 abort on key with confirm
*  1.5.2 track, sectors begin at 1 instead of 0
*        retrys on writes retry format-write-verify loop
*     on write errors, ESC to go to another disk
```

```
 *  1.5.1 port to megamax
 *     uses all memory
 * v 1.5: more checking of entry syntax.
 *        on-line help
 *        fills bad sectors with 0s before writing
 *     more sensible memory allocation
 *     s option for auto skip
 * v 1.4: one automatic retry without prompting. If skips, copies good
 *     sectors anyway.
 * v 1.3: when error on a track, asks for retrys, which are done sector
 *        by sector.
 * v 1.2: when double sided, copy the two sides
 * v 1.1: whole side in memory for one drive use.
 * v 1.0: original one in mark williams. quick and really dirty.
 */

/* ON_LINE_MANUAL:
 *================
 */

char           *help_text[] = {
"",
"                  FLIP 2.7 manual",
"                   Colas Nahaboo",
"                   colas@mirsa.inria.fr",
"",
"    flip - disk copier with error-recovery, on either sides.",
"           with FAST formatting as TWISTER or DCFORMAT",
"",
"SYNTAX",
"  the commands are given as: (items in [] are optional)",
"",
"      <source-drive>[<dest-drive>][fvVtios9]                  : copy disks",
"      f[<drive>][fvVtis9]                                     : format disk",
"      v[<drive>][tios9]                                       : verify disk",
"      r[<drive>][fvVi]                                : re-write disk in memory",
"      q                                                       : quits",
"      ?                                                       : help",
"",
"  A drive is specified by the letter \"a\" or \"b\", followed optionally by",
"  \"1\" or \"2\" to indicate the side. (default drive is \"a\", side is \"1\")",
"",
"OPTIONS",
"     f - to format destination disk (default don't)",
"     V - verify disk by re-reading track just after being formatted.",
"     v - verify copy by re-reading track just after being written",
"     t - to specify what track(s) to copy (used with \"f\" to give the",
"           number of tracks of the new disk, defaults to 80, e.g: say fft",
"           then give tracks 1 to 82 to format a disk with 82 tracks",
"           If you redo a command by hitting return, the tracks will be taken",
"           as the same, without re-prompting. (you can then format another",
"           82 track disk by hitting return in the above case).",
"           Note that the tracks need not be consecutive.",
```

```
"     i - interactive: ask user what to do in case of error (default don't)",
"           that is retry in track mode, go to sector mode or skip",
"     o - override: do not look at the source disk boot sector to determine",
"           its type, take 10 sectors, 1 sided, 80 tracks, unless told",
"           otherwise by options \"9\", \"d\" or \"t\"",
"     9 - disk is 9 sectors per track (default 10). Used with \"f\" or \"o\"",
"     s - disk is single-sided (default double). Used with \"f\" or \"o\"",
"",
"EXAMPLES",
"  ab2fv - to copy a non-protected SS disk in drive A on the unformatted",
"          back of a SS disk in drive B, verifying the copy.",
"  afv   - (or fv) to copy ANY disk with format and verify, in drive A.",
"  rfv   - to re-write last copy into drive A",
"  ffv   - to format and verify a double-sided disk in drive A, (10 sec/tr)",
"  fs    - to erase all files of a previously formatted single sided disk",
"          (only erase tracks 0 and 1)",
"",
"OPERATION",
"  During copy, flip prints for each track, a:",
"  ,    for reading track without problem",
"  ;    for reading track successfully after 4 retrys sector-by-sector",
"  -    for formatting a track (in FAST format)",
"  v    for verifying a track",
"",
"  You can pause during the copy at any time by pressing a key, but pressing",
"  ESC will abort. ^C always quits.",
"",
"  Without i option, on read error, it retries 4 times to read the track,",
"  then reads sector by sector. With option i, it prompts you for what to do:",
"  (the track and sector number given start at 1, but sector number at 0)",
"    - RETURN or SPACE retries. You'd better try to extract, then re-insert",
"      the disk to try to re-center it. (or try on another drive later)",
"    - \"s\" skips the bad sectors, going on with the rest of the copy.",
"      the bad sectors will be filled with 0s on the copy",
"      (but all that could be read is kept, of course!)",
"    - if it is writing the copy, you can press BACKSPACE if you want to try",
"      to write to another disk.",
"    - any other key aborts the copy.",
"  Experience shows that you can recover most of your damaged disks this way.",
"  If it cannot read the track, it reads it sector by sector, but tells you so",
"  because due to a bug in the TOS, a sector can be read incorrectly without",
"  giving an error. So FLIP does 4 reads of the same sector and compares",
"  them to be sure, but even if reads are equals, it warns you because the",
"  risk subsists, due to the bug in old TOS roms...",
"",
"DESCRIPTION",
"  The program copies TOS disks as a whole, and only non-protected",
"  ones. It can consider the back side of a single-sided disk as another disk",
"  so, it is very useful to use the back sides of protected disks, or",
"  single sided disks, such as MAGIC! ones. (with a two-side drive, of",
"  course!)",
"",
"  But it also reads disks with care, being able to read CORRECTLY most disks",
```

```
"  causing a read error to copiers or the GEM desktop. Thus, if you get a",
"  read error on a disk, copy it with FLIP!. Moreover, if it cannot read a",
"  sector, it tells you so! (Unlike the HELP program)",
"",
"  Version 2.6 can format and changes randomly the serial # of the disk",
"  when copying, (without the override option). It copies also executable",
"  boot sectors, but tells you so (in 2.6b) so that you can check for a VIRUS.",
"  it gives you the values at $3A offset to identify the virus",
"  Known virus values are: 41,FA,FF                                          ",
"                                                                            ",
"  When it reads a disk, it first tries to read the boot sector in order",
"  to know the size (tracks and sectors per track) of the disk. If it",
"  can't make sense out of it, it uses 80 tracks by 9 sector, unless you",
"  used the \"o\" option",
"  Thus you don't have to tell him the type of the disk you want to copy.",
"  It will recognize double sided disks automatically, but you can only",
"  copy double sided disks, without flipping their faces!",
"",
"  So, NEVER COPY A DISK WICH GIVES READ ERRORS WITH A BIT-COPIER (procopy)",
"  use FLIP instead!",
"",
"  From version 2.0 on, FLIP format in FAST format, all disks accesses on",
"  this disk will be twice faster compared to a normal disk!. It ALWAYS ",
"  formats disks in 10 sectors, even if only to store 9 sectors.",
"  And since version 2.7 I upgraded the TWISTER code so that now flip works on",
"  ALL ROM RELEASES!",
"",
"  Flip is not as fast as Procopy, because it formats the disk before writing",
"  on it, but is much safer, (I experienced a 90% failure rate with procopy",
"  when formatting and reading on different STs), and is as fast as Dcformat.",
"",
"COPYRIGHT",
"  This program is PUBLIC DOMAIN, and should be given with its sources",
"  Feel free to improve it!, but please mention my name.",
"  I would be very glad if you send me back any improved version, such as",
"  a GEM port",
"  CREDITS: Flip uses code from TWISTER from Dave SMALL & Dan MOORE.",
"",
"BUGS",
"  Not under GEM - yet!",
"  Cannot copy IBM boot sectors (use DCFORMAT)",
"",
"AUTHOR",
"  Colas NAHABOO",
"  383 ch du clos d'Embertrand",
"  06250 MOUGINS",
"  FRANCE.",
"  Personal phone: 93 75 68 29, give me a call...",
"  Electronic mail address:    colas@mirsa.inria.fr",
0L
};

#include "osbind.h"
```

```
#include "stdio.h"

more(text)
char          **text;
{
    char          **p = text, c;
    int             line = 0;

    while (*p) {
     printf("%s\n", *p++);
     line++;
     if (line > 22) {
         printf("%cp more (SP, CR, BS or ESC) %cq", 27, 27);
         fflush(stdout);
         while (!Cconis());
         c = Cnecin();
         printf("%cM", 27);
         fflush(stdout);
         switch (c) {
         case 27:
         case 'q':
          printf("\n");
          return;
          break;
         case ' ':
          line = 0;
          break;
         case 8:
          line = 0;
          p -= 48;
          if (p < text)
              p = text;
          printf("%cp...Backing 1 page...%cq\n", 27, 27);
          break;
         }
     }
    }
    printf("\n");
}

@\Rogue\Monster\
else
  echo "shar: Will not over write MAN.C"
fi
if `test ! -s TWST.C`
then
echo "x - TWST.C"
cat > TWST.C << '@\Rogue\Monster\'
/*
 *       Megamax C inline ASM version of Twister.
 *       Converted from AS68 format  dlm  12/14/86.
 *       AS68 version by David Small before that date.
 *       Used for the MegAMin HD backup program.
```

```
 *
 */

/* patched by colas for use with FLIP:
 * add params:
 *   flipped: 0= no, 1=use back as front (must be SS then)
 *   endtrack: 80 for normal disks
 *   starttrack: 0 normally
 * outputs a '-' for each formatted cylinder (see  PUTCHAR comments)
 * no more verify
 */

/* routine that are callable from C are: */
extern twister();
extern confirm_abort();
extern printf();

#include "osbind.h"
#include "stdio.h"

/* global variables */
int thedisk, dblsided, sectoroffset;   /* input parameters */
int flipped, endtrack, starttrack, keypress, new_roms;     /* colas params */
long buffer;

int badflag;                          /* bad sector error flag */

/* local variables */
static int theside, thetrack, twistsectorno, sectorno, retrycnt,
          cdev, ctrack, csect, cside, ccount, A7t, interlv,
          virgin, def_error, curr_err;
static long cdma, edma, tmpdma, saveA2;
static int saveA0, saveA1, saveD0;
/* now the defines (AS68 equ's) */

/*   tunable values */
#define retries   2         /* default # of retries -1 */
#define midretry  1         /* "middle" retry, when to reseek  */
#define timeout   0x40000   /* short timeout, motor already on */
#define ltimeout  0x60000   /* long timeout, to startup motor
     */

/*   Floppy state variables in DSB: RAM usage */
#define recal    0xff00     /* recalibrate flag
     */
#define dcurtrack 0         /* current track number
     */
#define dseekrt   dcurtrack+2 /* seek rate     */
#define dsbsiz    dseekrt+2 /* size of a DSB   */

/* hardware equates */
/* DMA chip        */
#define diskctl    0xffff8604    /* disk controller data access */
```

```
#define fifo       0xffff8606    /* DMA mode control / status
      */
#define dmahigh    0xffff8609    /* DMA base high
      */
#define dmamid     0xffff860b    /* DMA base mid
      */
#define dmalow     0xffff860d    /* DMA base low
      */

/* 1170 select values  */
#define cmdreg     0x80          /* select command register
     */
#define trkreg     0x82          /* select track register
     */
#define secreg     0x84          /* select sector register
     */
#define datareg    0x86          /* select data register
     */

/* GI sound chip (drive a/b select lines and side select) */
#define giselect   0xffff8800    /* (W) sound chip register select */
#define giread     0xffff8800    /* (R) sound chip read data
     */
#define giwrite    0xffff8802    /* (W) sound chip write data
     */
#define giporta    0x0e          /* GI register # for i/o port A */

/* 68901 mfp chip */
#define mfp        0xfffffa00    /* mfp base             */
#define gpip       mfp+1         /* general purpose i/o
     */

/* misc defines */
#define seekrate 3
#define dsb0old  0xa06
#define dsb0new  0xa4c      /* for the Mega ROMs */
#define dsb1old  0xa0a
#define dsb1new  0xa50
#define flock   0x43e

/* ROM burn dates */
#define old_rom  0x11201985L /* french = 0x04241986L */
#define new_rom  0x04221987L

#define rom_date 0x00fc0018L         /* rom burn date address */

char *abort_message = "\n\007KEY PRESSED! CR/SP = continue, other abort:\n";

asm {         /* start of inline ASM */

/*
 *   All the following is munged code from AS68 (that's why the
 *   weird format).  The AS68 code was munged from the BIOS
```

```
 *   source.  And most of this is from the FDC article in Start
 *   issue number 3.
 *
 *   Anyway, we apologize for the mess, but heh it works.
 *
 *   Oh yes.  The labels are usually on nops, not empty lines.  Thats
 *   because AS68 occasionally pukes on labels on empty lines.  And
 *   I didn't take the time to remove them.  (Megamax doesn't care.)
 *
 */

;*************************************************************
; Twister: Faster formatter (zipformat) hack.  MM version
        *
; Copyright (c) 1986, 1987 START Magazine (as usual)
        *
; Written by David Y. Small & Dan Moore                    *
;*************************************************************
;
;              Dedication:
;
;     For Bill The Cat: because he's twisted, too.
;
;*************************************************************
;
; code begins.
;
;  PASS ME THIS STUFF DAN! Then SHAVE! For God's sakes! What
; are you trying to do, look like Fidel Castro?
;
; And get SOME BEER in here okay? Sheesh!
;
; input integers (if coming from C):
;
; dblsided: 0000 or ffff
; thedisk: 0 = a, 1 = b
;*************************************************************


;*******************************************************
; Frammatter
;*******************************************************
twister:  move.l   A2, saveA2(A4)     ; everything else saved by the Supexec

        clr.w      badflag(A4)        ; assume no errors
;
;*** Init vars for formatting loop
;
     move.w D0, saveD0(A4);
     move.w starttrack(A4), D0   ;start at starttrack
     move.w D0, thetrack(A4)
     move.w saveD0(A4), D0
;WAS    move.w  #0,thetrack(A4)   ; start at track 0
```

```
        move.w  #1,twistsectorno(A4) ; twister value

     move.w A0, saveA0(A4)
     move.w A1, saveA1(A4)
trackloop: nop                ; comes here for subseq tracks
     move.w saveA0(A4), A0
     move.w saveA1(A4), A1

;***** FRONT SIDE ***********
;**** Setup stack: .. based on read sec command, tweaked for frammat

        move.w  #0xe5e5,-(A7)    ; virgin data -- e5's work okay
        move.l  #0x87654321,-(A7) ; magic # to make format work / fmt only
        move.w  #1,-(A7)      ; sector interleave factor / fmt only
        move.w  flipped(A4),-(A7)       ; ** side.w (relevant) / side: FRONT
        move.w  thetrack(A4),-(A7)  ; ** track.w (relevant) / same
        move.w  #10,-(A7)     ; sector.w (irrelevant) / sectors per track
        move.w  thedisk(A4),-(A7)      ; ** device.w  (relevant) / same
        move.l  #0,-(A7)      ; dummy.l (irreleveant) / same
        move.l  buffer(A4),-(A7) ; ** buffer address.l (relevant) / same
        jsr     _flopfmt
        adda.l  #24,A7        ; Fix stack.
        cmpi.w  #0xffff, badflag(A4)   ; aborts on error
        beq     gemexit
;****
; Jazz twistsectorno(A4). Look at what it ended up as; set it to
; the next track twist depending on that.
;
; It will end up as "+1" from the last sector formatted, because
; the formatter bumps it up.
;
        cmpi.w  #1,twistsectorno(A4)      ; 0-1 transition
        beq     fgoto1
        cmpi.w  #9,twistsectorno(A4)      ; 1-2 transition
        beq     fgoto2
        cmpi.w  #7,twistsectorno(A4)      ; 2-3 transition
        beq     fgoto3
        cmpi.w  #5,twistsectorno(A4)      ; 3-4 transition
        beq     fgoto4
        cmpi.w  #3,twistsectorno(A4)      ; 4-0 transition
        beq     fgoto0

        dc.b  0x4a
        dc.b  0xfc               ; oops, die (illegal instruction)
;*****
; Really class coding, eh? Oh,well, it's easy and it works, which
; twelve other fancy ways of doing this *don't*.
;
fgoto1: move.w  #0x9,twistsectorno(A4)         ; 9,a,1,2,3,4,5,6,7,8
        bra     fkeepformatting
fgoto2: move.w  #0x7,twistsectorno(A4)         ; 7,8,9,a,1,2,3,4,5,6
        bra     fkeepformatting
```

```
fgoto3: move.w  #0x5,twistsectorno(A4)         ; 5,6,7,8,9,a,1,2,3,4
        bra     fkeepformatting
fgoto4: move.w  #0x3,twistsectorno(A4)         ; 3,4,5,6,7,8,9,a,1,2
        bra     fkeepformatting
fgoto0: move.w  #0x1,twistsectorno(A4)         ; 1,2,3,4,5,6,7,8,9,a
        bra     fkeepformatting
        nop
;*****
fkeepformatting: nop
;*****
; now, do a verify on those sectors.
;**** Setup stack: .. based on read sec command, tweaked for frammat
;
;     tst.w    sectoroffset(A4)  ; no verify if 11 to 20 sector
;     ; numbering
;     bne  nover1
;
;     move.w  #0xa,-(A7)        ; count.w 2
;     move.w  flipped(A4),-(A7)    ; ** side.w (relevant) / side: FRONT 4
;     move.w  thetrack(A4),-(A7)  ; ** track.w (relevant) / same 6
;     move.w  #0x1,-(A7)        ; sector.w (relevant) / starting sector 8
;     move.w  thedisk(A4),-(A7)   ; ** device.w  (relevant) / same 0xa
;     move.l  #0,-(A7)    ; dummy.l (irreleveant) / same 0xe
;     move.l  buffer(A4),-(A7) ; ** buffer address.l (relevant) / same 0x12
;
;     move.w  #19,-(A7)   ; do BIOS _flopver 0x14
;     trap     #14
;     adda.l  #20, A7         ; fix stack.
;
;     tst.l    D0
;     bne  badsecs
;
;****

;***** BACK SIDE ******     (How kinky Dave.  What _are_ you talking about?)

nover1:  tst.w   dblsided(A4)
         beq     nobackside

;**** Setup stack: .. based on read sec command, tweaked for frammat

        move.w  #0xe5e5,-(A7)    ; virgin data -- e5's work okay
; and with a virgin.  My, you really are weird Dave.
        move.l  #0x87654321,-(A7) ; magic # to make format work / fmt only
        move.w  #1,-(A7)       ; sector interleave factor / fmt only
        move.w  #1,-(A7)       ; ** side.w (relevant) / side: BACK
        move.w  thetrack(A4),-(A7)  ; ** track.w (relevant) / same
        move.w  #10,-(A7)      ; sector.w (irrelevant) / sectors per track
        move.w  thedisk(A4),-(A7)      ; ** device.w  (relevant) / same
        move.l  #0,-(A7)       ; dummy.l (irreleveant) / same
        move.l  buffer(A4),-(A7) ; ** buffer address.l (relevant) / same
        jsr     _flopfmt
        adda.l  #24,A7         ; Fix stack.
```

```
        cmpi.w  #0xffff, badflag(A4)   ; aborts on error
        beq     gemexit
;*****
;
; Jazz twistsectorno(A4). Look at what it ended up as; set it to
; the next track twist depending on that.
;
; It will end up as "+1" from the last sector formatted, because
; the formatter bumps it up.
;
        cmpi.w  #1,twistsectorno(A4)      ; 0-1 transition
        beq    goto1
        cmpi.w  #9,twistsectorno(A4)      ; 1-2 transition
        beq    goto2
        cmpi.w  #7,twistsectorno(A4)      ; 2-3 transition
        beq    goto3
        cmpi.w  #5,twistsectorno(A4)      ; 3-4 transition
        beq    goto4
        cmpi.w  #3,twistsectorno(A4)      ; 4-0 transition
        beq    goto0

        dc.b   0x4a
        dc.b   0xfc                ; oops, die

;
goto1:  move.w  #0x9,twistsectorno(A4)         ; 9,a,1,2,3,4,5,6,7,8
        bra    keepformatting
goto2:  move.w  #0x7,twistsectorno(A4)         ; 7,8,9,a,1,2,3,4,5,6
        bra    keepformatting
goto3:  move.w  #0x5,twistsectorno(A4)         ; 5,6,7,8,9,a,1,2,3,4
        bra    keepformatting
goto4:  move.w  #0x3,twistsectorno(A4)         ; 3,4,5,6,7,8,9,a,1,2
        bra    keepformatting
goto0:  move.w  #0x1,twistsectorno(A4)         ; 1,2,3,4,5,6,7,8,9,a
        bra    keepformatting
        nop
;
keepformatting: nop           ; entry after twisting.
;
;
;     ;**** Setup stack: .. based on read sec command, tweaked for frammat
;
;     tst.w    sectoroffset(A4) ; no verify for strange sector numbers
;     bne  nobackside
;
;     move.w  #0xa,-(A7)        ; count.w 2
;     move.w  #1,-(A7)    ; ** side.w (relevant) / side: BACK 4
;     move.w  thetrack(A4),-(A7)  ; ** track.w (relevant) / same 6
;     move.w  #0x1,-(A7)        ; sector.w (relevant) / starting sector # 8
;     move.w  thedisk(A4),-(A7)   ; ** device.w  (relevant) / same 0xa
;     move.l  #0,-(A7)    ; dummy.l (irreleveant) / same 0xe
;     move.l  buffer(A4),-(A7) ; ** buffer address.l (relevant) / same 0x12
;
```

```
;     move.w  #19,-(A7)   ; do BIOS _flopver 0x14 (why not)
;     trap     #14
;     adda.l  #20, A7          ; fix stack
;
;     tst.l    D0
;     bne  badsecs        ; wham
;
;****
nobackside: nop      ; entry if not DS  (well at least this isn't kinky Dave)

; add 1 to thetrack. If endtrack, quit.
        addi.w   #1, thetrack(A4)
     move.w A0, saveA0(A4)
     move.w A1, saveA1(A4)

     move.w D0, saveD0(A4)       /* PUTCHAR */
     move.w #0x2D, -(A7)         /* outputs a '-' */
     move.w #6, -(A7)
     trap #1
     adda.l #4, A7
     move.w #0xB, -(A7)      /* CNONIS */
     trap #1
     adda.l #2, A7
     cmpi.w  #0,D0
         bne key_pressed              /* aborts on keypress */
resume:  move.w saveD0(A4), D0       /* PUTCHAR end */

     move.w endtrack(A4), A0
     move.w thetrack(A4), A1
     cmpa A0, A1
        bne    trackloop      ; "do another"

     move.w saveA0(A4), A0
     move.w saveA1(A4), A1
        bra    gemexit        ; Continue, etc, and so forth

;*********
key_pressed:
     MOVE.w   #8,-(A7)
     trap #1
     ADDa.L   #2,A7
     CMPI.w   #27,D0
     beq  key_abort
     move.l  abort_message(A4),-(A7)
     move.w  #9,-(A7)
     trap #1
     ADDa.L   #6,A7
     MOVE.w   #8,-(A7)
     trap #1
     ADDa.L   #2,A7
     CMPI.w   #32,D0
     beq  resume
     CMPI.w   #10,D0
```

```
     beq  resume
     CMPI.w   #13,D0
     beq  resume
key_abort:    MOVE.w   #1,keypress(A4)
     move.w  saveD0(A4), D0
     bra gemexit

;*********
badsecs:  move.w #0xffff, badflag(A4)  ; bad sectors found

;**************
;
; End. Exit to calling routine.
;
gemexit:  movea.l saveA2(A4), A2  ; only reg not saved by Supexec
          rts

;*****************************************************
;
;-------------------------------------------------------------------------
;       130-ST / 520-ST
;       Floppy Disk Driver
;       (C)1985 Atari Corp.
;       From the FDC article (Start issue 3); shortened to just
;       the formatter routine and the low level I/O.

;
;-------------------------------------------------------------------------
;
;*************************************************************
;
; _flopfmt - format a track
; Passed (on the stack):
;       0x1a(A7) initial sector data
;       0x16(A7) magic number
;       0x14(A7) interleave
;       0x12(A7) side
;       0x10(A7) track
;        0xe(A7) A7t(A4)
;        0xc(A7) drive
;        0x8(A7) pointer to state block
;        0x4(A7) dma address
;        0x0(A7) [return]
;
; Returns:     EQ: track successfully written.  Zero.W-terminated list of
;              bad sectors left in buffer (they might /all/ be bad.)
;
;              NE: could not write track (write-protected, drive
;              failure, or something catastrophic happened).
;-
_flopfmt:
        cmpi.l   #0x87654321, 0x16(A7)        ; check for magic# on stack
        bne    flopfail          ; no magic, so we just saved the world
```

```
;       bsr     change            ; check for disk flip
;       moveq   #e_error,D0       ; set default error number
        bsr     floplock              ; lock floppies, setup parms
        bsr     select                ; select drive and side
        move.w  0xe(A7),A7t(A4)       ; save sectors-per-track
        move.w  0x14(A7),interlv(A4)      ; save interleave factor
        move.w  0x1a(A7),virgin(A4)       ; save initial sector data

;--- put drive into "changed" mode
;       moveq   #m_changed,D0         ; D0 = "CHANGED"
;       bsr     setdmode              ; set media change mode

;--- seek to track (hard seek):
;debug* move.l  #0xff00ff00,0xf8030
        bsr     hseek                 ; hard seek to 'ctrack'
;debug* move.l  #0x00000000,0xf8030
        bne     flopfail              ; (return error on seek failure)
        move.w  ctrack(A4),dcurtrack(A1)    ; record current track#

;--- format track, then verify it:
;*      move.w  #e_error,curr_err(A4)(A5)   ; vanilla error mode
;debug* move.l  #0xff00ff00,0xfA030
        bsr     fmtrack               ; format track
;debug* move.l  #0x0,0xfA030

        bne     flopfail              ; (return error on seek failure)
        bra     flopok


;****************************
;+
; fmtrack - format a track. Tweaked for skewed interleave.
;
; Passed:       variables setup by _flopfmt
; Returns:      NE on failure, EQ on success
; Uses:         almost everything
; Called-by:    _flopfmt
;
;-
fmtrack:
;*      move.w  #e_write,def_error(A4)    ; set default error number
        move.w  #1,D3       ; start with sector 1, first pass
;**
        movea.l  cdma(A4),A2          ; A2 =  prototyping area
; Lay down beginning of track

        move.w  #60-1,D1              ; 60 x 0x4e (track leadin)
        move.b  #0x4e,D0
        bsr     wmult
; Repeat 10 times: sector data.
        move.w  #1,sectorno(A4)           ; how many secs written
; Note that twistsectorno is initialized out of this routine.
;--- address mark
```

```
secloop: nop
;
ot1:    move.w  #12-1,D1             ; 12 x 0x00
        clr.b   D0
        bsr     wmult
        move.w  #3-1,D1              ; 3 x 0xf5
        move.b  #0xf5,D0
        bsr     wmult
        move.b  #0xfe,(A2)+           ; 0xfe -- address mark intro
        move.b  ctrack+1(A4),(A2)+       ; track# - low half of word
        move.b  cside+1(A4),(A2)+            ; side# - low half of word

;* #1: just use a plain sector number
;*      move.b  D4,(A2)+             ; sector#

;* #2: use a twisted sector number
;*      move.b  twistsectorno+1(A4),(A2)+   ; new: sector #.

        move.w  twistsectorno(A4),D0      ; fetch    1-10 sector #
;
; Note that meg-a-minute backup uses sectors # 11-20 to force GEM
; not to use MM disks.
;
        add.w   sectoroffset(A4),D0   ; add possible shift to 11-20 sector #
        move.b  D0,(A2)+              ; plug it into sector table

; Add 1 to sector #. If it is b, wrap it to 1.
        addi.w  #1,twistsectorno(A4)      ; real sector # being plopped
        cmpi.w  #0xb,twistsectorno(A4)
        bne     notb
        move.w  #1,twistsectorno(A4)
;
notb:   nop
        move.b  #0x02,(A2)+           ; sector size (512)
        move.b  #0xf7,(A2)+           ; write checksum

;--- gap between AM and data:
        move.w  #22-1,D1             ; 22 x 0x4e
        move.b  #0x4e,D0
        bsr     wmult
        move.w  #12-1,D1             ; 12 x 0x00
        clr.b   D0
        bsr     wmult
        move.w  #3-1,D1              ; 3 x 0xf5
        move.b  #0xf5,D0
        bsr     wmult

;--- data block:
        move.b  #0xfb,(A2)+           ; 0xfb -- data intro
        move.w  #256-1,D1        ; 256 x virgin.w (initial sector data)
ot2:    move.b  virgin(A4),(A2)+         ; copy high byte
        move.b  virgin+1(A4),(A2)+       ; copy low byte
        dbf     D1,ot2               ; fill 512 bytes
```

```
        move.b  #0xf7,(A2)+           ; 0xf7 -- write checksum
        move.w  #40-1,D1             ; 40 x 0x4e
        move.b  #0x4e,D0
        bsr     wmult
;
; Next sector, or, end-of-track. If we hit sector 11, time to quit.
;
        addi.w   #1,sectorno(A4)
        cmpi.w  #11,sectorno(A4)
        bne     secloop        ; loop again
;
; Okay, 10 sectors laid down.
;
;--- end-of-track
        move.w  #1400,D1         ; 1401 x 0x4e -- end of track trailer
        move.b  #0x4e,D0
        bsr     wmult

;--- setup to write the track:
        move.b  cdma+3(A4),dmalow      ; load dma pointer
        move.b  cdma+2(A4),dmamid
        move.b  cdma+1(A4),dmahigh
        move.w  #0x190,(A6)            ; toggle R/W flag and
        move.w  #0x090,(A6)            ; select sector-count register
        move.w  #0x190,(A6)
        move.w  #0x1f,D7               ; (absurd sector count)
        bsr     wdiskctl
        move.w  #0x180,(A6)            ; select 1770 cmd register
        move.w  #0xf0,D7               ; write format_track command
;debug* move.l  #0xff00ff00,0xfc030
        bsr     wdiskctl
        move.l  #timeout,D7           ; D7 = timeout value

;--- wait for 1770 to complete:
otw1:   btst.b  #5,gpip               ; is 1770 done?
        beq     otw2                  ; (yes)
        subq.l  #1,D7                 ; if(--D7) continue;
        bne     otw1
;debug* move.l  #0xffffffff,0xfc030

        bsr     reset1770             ; timed out -- reset 1770
oterr:  moveq  #1,D7                  ; return NE (error status)
        rts

;--- see if the write-track won:
otw2:   nop
;debug* move.l  #0x0,0xfc030

        move.w  #0x190,(A6)            ; check DMA status bit
        move.w  (A6),D0
        btst    #0,D0             ; if its zero, there was a DMA error
        beq     oterr                 ; (so return NE)
        move.w  #0x180,(A6)            ; get 1770 status
```

```
        bsr     rdiskctl
;**     bsr     err_bits                ; set 1770 error bits
        and.b   #0x44,D0  ; check for writeProtect & lostData
        rts                     ; return NE on 1770 error

;------ write 'D1+1' copies of D0.B into A2, A2+1, ...
wmult:  move.b  D0,(A2)+                ; record byte in proto buffer
        dbf     D1,wmult                ; (do it again)
        rts
;
;*************************************************************
;
; floplock - lock floppies and setup floppy parameters
;
; Passed (on the stack):
;       0x18(A7) - count.W (sector count)
;       0x16(A7) - side.W (side#)
;       0x14(A7) - track.W (track#)
;       0x12(A7) - sect.W (sector#)
;       0x10(A7) - dev.W (device#)
;        0xc(A7) - obsolete.L
;        8(A7) - dma.L (dma pointer)
;        4(A7) - ret1.L (caller's return address)
;        0(A7) - ret.L (floplock's return address)
;
; It is not coincidence that this matches the read and write input list;
; they call us to get stuff off stack and plug into parameters. Then,
; routine "select" actually pops this stuff into hardware.
;
; Passed:       D0.W = default error number
;
; Also, we helpfully point A6 at the DMA chip, and A1 at the DSB.
;-
floplock:
        lea    regsave, A0
        movem.l D3-D7/A3-A6, (A0)          ; save C registers
        lea     fifo,A6                 ; A6 -> fifo
; Start setting up param block..
        move.w  D0,def_error(A4)           ; set default error number
        move.w  D0,curr_err(A4)            ; set current error number
; Kick VBL off floppies..
        move.w  #1,flock                ; tell vbl not to touch floppies
; Stuff off stack:
        move.l  8(A7),cdma(A4)             ; cdma -> /even/ DMA address
        move.w  0x10(A7),cdev(A4)        ; save device# (0 .. 1)
        move.w  0x12(A7),csect(A4)    ; save sector# (1 .. 9, usually)
        move.w  0x14(A7),ctrack(A4)   ; save track# (0 .. 39 .. 79 ..)
        move.w  0x16(A7),cside(A4)    ; save side# (0 .. 1)
        move.w  0x18(A7),ccount(A4)   ; save sector count (1..A)
;--- pick a DSB: Point A1 at it.
; we need to check which ROMs also, if old use that
; otherwise assume it is a new machine
        lea     dsb0new,A1          ; pick dsb 0 (drive A)
```

```
        cmpi.w  #0, new_roms(A4)
        bne     rom1
        lea   dsb0old,A1
rom1:   tst.w   cdev(A4)
        beq     flock2
         lea     dsb1new,A1         ; pick dsb 1 (drive B)
         cmpi.w  #0, new_roms(A4)
         bne     flock2
         lea     dsb1old,A1
;--- compute ending DMA address from count parameter: Plug into edma.
; This is used in multisector transfers in read-multiple-sector,
; but we don't use it here.
flock2: moveq  #0,D7
        move.w  ccount(A4),D7            ; edma = cdma + (ccount * 512)
        lsl.w  #8,D7
        lsl.w  #1,D7                  ; do a 9 shift..
        movea.l  cdma(A4),A0
        adda.l   D7,A0
        move.l  A0,edma(A4)               ; save in edma
;--- recalibrate drive, if it needs it. This only happens when the DSB
; says that this drive has never awoken before, and needs an initial
; recal to match its current track # with the DSB track #.
        tst.w  dcurtrack(A1)          ; if (curtrack < 0) recalibrate()
        bpl    flockr
; RECAL needed. Show flag about it..
        move.l  #0xf0f0f0f0,0xfe000
        move.l  #0xf0f0f0f0,0xfe004

        bsr    select                 ; select drive & side
        clr.w  dcurtrack(A1)          ; we're optimistic -- assume winnage
; Restore.
        bsr    restore                ; attempt restore
        beq    flockr                 ; (it won)
        moveq  #10,D7                 ; attempt seek to track 10
        bsr    hseek1
        bne    flock1                 ; (failed)
        bsr    restore                ; attempt restore again
        beq    flockr                 ; (it won)
;
flock1:  move.w  #recal, dcurtrack(A1)      ; complete failure (what can we do?)

        movem.l regsave(PC), D3-D7/A3-A6
flockr: rts
;
;*********************************************************************
;+
; flopfail - unlock floppies and return error.
; Common way for read and write to return.
;
; Note: Returns via unlok1.
;-
flopfail:
        move.w #0xffff, badflag(A4)           ; bad sectors found
```

```
        move.l  #0xffffffff,D0         ; aargh, error
        bra   unlok1

;************************************************
;+
; flopok - unlock floppies and return success status. Also a common
; way for r/w to return.
;
;-
flopok: clr.l  D0                     ; return 0 (success)
; Entry point from flopfail..
unlok1: move.l  D0,-(A7)              ; (save return value)
; I believe this code returns the FDC's status to a type-1 status, where
; the write protect switch is available for VBL to look at.
        move.w  #datareg,(A6)         ; force WP to real-time mode
; Sets FDC's current track register to track we are on right now.
        move.w  dcurtrack(A1),D7      ; dest-track = current track
        bsr     wdiskctl
; Does it with a "noop seek" (source=dest). Only forces FDC to type 1 status.
        move.w  #0x10,D6               ; cmd = seek w/o verify
        bsr     flopcmds              ; do it
;
unlok2: move.l  (A7)+,D0              ; restore return value
        movem.l  regsave(PC), D3-D7/A3-A6
; clear floppy lock of vblank..
        clr.w   flock        ; allow vblank .. unlock floppies
; wave byebye at the pretty camera, jenny...
        rts
;************************ Seek routines ********************
;+
; hseek  - seek to 'ctrack(A4)' without verify
; hseek1 - seek to 'D7' without verify
; hseek2 - seek to 'D7' without verify, keep current error number
;
; Returns:     NE on seek failure ("cannot happen"?)
;              EQ if seek wins
;
; Uses:        D7, D6, ...
; Exits to:    flopcmds
; Called-by:   _flopfmt, _flopini
;
;-
hseek:  move.w  ctrack(A4),D7            ; dest track = 'ctrack'
hseek1: nop
hseek2: move.w  #datareg,(A6)         ; write destination track# to data reg
        bsr     wdiskctl              ; write D7 to FDC data register
; seek command:
        move.w  #0x10,D6               ; execute "seek" command
;                                     * Note: no spinup time.
        bra     flopcmds              ; (without verify...)
;
;*********************************************
;+
```

```
; reseek - home head, then reseek track
; Returns:     EQ/NE on success/failure
; Falls-into:  go2track
;
;-
reseek:
        bsr     restore                 ; restore head
        bne     go2trr                  ; (punt if home fails)

        clr.w   dcurtrack(A1)           ; current track = 0
        move.w  #trkreg,(A6)            ; set "current track" reg on 1770
        clr.w   D7
        bsr     wdiskctl                ; write a 00 to trk register.

        move.w  #datareg,(A6)           ; seek out to track five
        move.w  #5,D7
        bsr     wdiskctl                ; dest track = 5
        move.w  #0x10,D6
        bsr     flopcmds                ; seek
;                                       * Note: no spinup time.
        bne     go2trr                  ; return error on seek failure
        move.w  #5,dcurtrack(A1)        ; set current track#

;+++++++++++++++++++++++++++++++++++++++++++++++
; go2track - seek proper track
; Passed:      Current floppy parameters (ctrack, et al.).
; Returns:     EQ/NE on success/failure
; Calls:       flopcmds
; Called by: read sector, for instance. Lots of places.
;-
go2track:
        move.w  #datareg,(A6)           ; set destination track# in
        move.w  ctrack(A4),D7               ;  1770's data register
        bsr     wdiskctl                ; (write track#)
        moveq   #0x14,D6                 ; execute 1770 "seek_with_verify"
        bsr     flopcmds                ; (include seek-rate bits)
        bne     go2trr                  ; return error on seek failure
        move.w  ctrack(A4),dcurtrack(A1)    ; update current track number
        and.b   #0x18,D7                 ; check for RNF, CRC_error, lost_data
go2trr: rts                             ; return EQ/NE on succes/failure

;+++++++++++++++++++++++++++++++++++++++++++++++
;+
; restore - home head
; Passed:      nothing
; Returns:     EQ/NE on success/failure
;-
restore:
        clr.w   D6                      ; 0x00 = 1770 "restore" command
        bsr     flopcmds                ; do restore
        bne     res_r                   ; punt on timeout
        btst    #2,D7                   ; test TRK00 bit
        eori.w  #0x04, CCR              ; flip Z bit (return NE if bit is zero)
```

```
        bne     res_r                   ; punt if didn't win
        clr.w   dcurtrack(A1)           ; set current track#
res_r:  rts

;*****************************************************
; Special floppy cmd just for seeking:
;+
; flopcmds - floppy command (or-in seek speed bits from database)
; Passed:       D6.w = 1770 command
; Sets-up:      seek bits (bits 0 and 1) in D6.w
; Falls-into:   flopcmd
; Returns:      EQ/NE on success/failure
;
; I get the impression this is only used for seeking. I am right.
;-
flopcmds:
        move.w  dseekrt(A1),D0          ; get floppy's seek rate bits
        andi.b  #3,D0                   ; OR into command
        or.b    D0,D6
; Fall in...
;*****************************************************+
; flopcmd - execute any ol' 1770 command (with timeout)
; Passed:       D6.w = 1770 command
;
; Returns:      EQ/NE on success/failure
;               D7 = 1770 status bits
;
; Note: does motor spinup, if cmd in D6 specs it (I would *hope*!)
;-
flopcmd:
        move.l  #timeout,D7             ; setup timeout count (assume short)
        move.w  #cmdreg,(A6)            ; select 1770 command register
        bsr     rdiskctl                ; read it to clobber READY status
        btst    #7,D0                   ; is motor on?
        bne     flopcm                  ; (yes, keep short timeout)
        move.l  #ltimeout,D7            ; extra timeout for motor startup
flopcm: bsr     wdiskct6                ; write command (in D6)

flopc1: subq.l  #1,D7                   ; timeout?
        beq     flopcto                 ; (yes, reset and return failure)
        btst.b  #5,gpip                 ; 1770 completion?
        bne     flopc1                  ; (not yet, so wait some more)
        bsr     rdiskct7                ; return EQ + 1770 status in D7
        clr.w   D6
        rts
;****
flopcto:
; We timed out.. a gruesome death indeed.
; Whap controller back to life.  (Hit him harder Dave, he likes it that way)
        bsr     reset1770               ; bash controller
        moveq   #1,D6                   ; and return NE
        rts
```

```
;*****************************************************
;+
; reset1770 - reset disk controller after a catastrophe
; Passed:      nothing
; Returns:     nothing
; Uses:        D7
;-
reset1770:
        move.w  #cmdreg,(A6)          ; execute 1770 "reset" command
        move.w  #0xD0,D7               ; force interrupt
        bsr     wdiskctl
        move.w  #15,D7                ; wait for 1770 to stop convulsing
r1770:  dbf     D7,r1770              ; (short delay loop)
        bsr     rdiskct7              ; return 1770 status in D7
        rts

;*****************************************************
;+
; select - setup drive select, 1770 and DMA registers
; Passed:      cside(A4), cdev(A4)
; Returns:     appropriate drive and side selected
;
; Called: All over the place.
;
;-
select: move.w  #0,0x9c0               ; floppies NOT deselected
;                                     * lets VBL shut them off...
        move.w  cdev(A4),D0               ; get device number
        addq.b  #1,D0                 ; add and shift to get select bits
        lsl.b   #1,D0                 ; into bits 1 and 2
        or.w    cside(A4),D0              ; or-in side number (bit 0)
        eori.b   #7,D0                 ; negate bits for funky hardware select
        andi.b   #7,D0                 ; strip anything else out there
        bsr     setporta              ; do drive select
; Have to restore 1770's track register from table..in dsb.
        move.w  #trkreg,(A6)          ; setup 1770 track register
        move.w  dcurtrack(A1),D7      ; from current track number
        bsr     wdiskctl
        clr.b   tmpdma(A4)                ; zero bits 24..32 of target DMA addr

; Setup R/W parameters on 1770. Used by
; r/w sector, among others. This is where the sector register gets set.
;
select1:
        move.w  #secreg,(A6)          ; setup requested sector_number from
        move.w  csect(A4),D7              ;       caller's parameters
        bsr     wdiskctl
        move.b  cdma+3(A4),dmalow             ; setup DMA chip's DMA pointer
        move.b  cdma+2(A4),dmamid
        move.b  cdma+1(A4),dmahigh
        rts

;*****************************************************
```

```
;+
; setporta - set floppy select bits in PORT A on the sound chip
; Passed:      D0.b (low three bits)
; Returns:     D1 = value written to port A
;              D2 = old value read from port A
; Uses:        D1
;-
setporta:
        move    SR,-(A7)              ; save our IPL
        ori     #0x0700,SR             ; start critical section
        move.b  #giporta,giselect         ; select port on GI chip
        move.b  giread,D1             ; get current bits
        move.b  D1,D2                 ; save old bits for caller
        andi.b   #0xff-7,D1             ; strip low three bits there
        or.b    D0,D1                 ; or-in our new bits
        move.b  D1,giwrite            ; and write 'em back out there
        move    (A7)+,SR              ; restore IPL to terminate CS, return
        rts

;************************************************
;+
; Primitives to read/write 1770 controller chip (DISKCTL register).
;
; The 1770 can't keep up with full-tilt CPU accesses, so
; we have to surround reads and writes with delay loops.
; This is not really as slow as it sounds.
;
wdiskct6:                             ; write D6 to diskctl
        bsr     rwdelay               ;       delay
        move.w  D6,diskctl            ;       write it
        bra     rwdelay               ;       delay and return

wdiskctl:                             ; write D7 to diskctl
        bsr     rwdelay               ;       delay
        move.w  D7,diskctl            ;       write it
        bra     rwdelay               ;       delay and return

rdiskct7:                             ; read diskctl into D7
        bsr     rwdelay               ;       delay
        move.w  diskctl,D7            ;       read it
        bra     rwdelay               ;       delay and return

rdiskctl:                             ; read diskctl into D0
        bsr     rwdelay               ;       delay
        move.w  diskctl,D0            ;       read it
; And here's the delay loop:
rwdelay:
        move    SR,-(A7)              ; save flags
        move.w  D7,-(A7)              ; save counter register
        move.w  #0x20,D7              ; 0x20 seems about right...
rwdly1: dbf     D7,rwdly1             ; busy-loop: give 1770 time to settle
        move.w  (A7)+,D7              ; restore register, flags, and return
        move    (A7)+,SR
```

```
       rts

;-------------------------------------
regsave:  dc.l     1      ; PC relative data area
          dc.l     2
          dc.l     3
          dc.l     4
          dc.l     5
          dc.l     6
          dc.l     7
          dc.l     8
          dc.l     9


}         /* MY GOD!  It's DONE!  */

@\Rogue\Monster\
else
  echo "shar: Will not over write TWST.C"
fi
if `test ! -s MAKEFILE`
then
echo "x - MAKEFILE"
cat > MAKEFILE << '@\Rogue\Monster\'
VERSION=27

flip_$(VERSION).tos:   flip.o twst.o man.o
     mmlink -o flip_$(VERSION).tos flip.o twst.o man.o

imp: flip.o twst.o man.o
     mmimp flip.o
     mmimp twst.o
     mmimp man.o
     mmlink -o flip_$(VERSION).tos flip.o twst.o man.o

@\Rogue\Monster\
else
  echo "shar: Will not over write MAKEFILE"
fi
if `test ! -s README`
then
echo "x - README"
cat > README << '@\Rogue\Monster\'
Flip is a small copier that I wrote for my personal use to:

- use the back of SS disks

- try to save data from damaged disks (getting frustrated with the abort
  message of DCFORMAT)

However, it evolved into (to my opinion) the BEST disk copier on the ST for
unprotected disks.
Since I and my friends are using it all the time, it might be of use to some
```

```
netlanders. As it was developed for my own use, it is not under GEM.
It is in no way as great a program as is GULAM or UNITERM

It is completly public domain, files are: (MEGAMAX C v1.1)

binaries:
     flip_26b.tos           the executable

sources:
     makefile
     flip.c        main code
     man.c         manual & history
     twst.c        formatter in in-line assembly, in fact TWISTER 2.0 code
               slightly modified (MEGAMAX v1.1)
               WORKS WITH ANY ROMS!
@\Rogue\Monster\
else
  echo "shar: Will not over write README"
fi
echo "Finished archive 2 of 2"
# to concatenate archives, remove anything after this line
exit 0


From 5283460342234291005
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,b14afce3f80d60cc,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-07-18 07:46:29 PST
Path: utzoo!attcan!uunet!lll-winken!lll-lcc!ames!mailrus!uwmcsd1!ig!agate!saturn!
ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i064: poplib -- shared libraries, with examples
Keywords: shar, MWC
Message-ID: <4194@saturn.ucsc.edu>
Date: 18 Jul 88 14:46:29 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 715
Approved: koreth@ssyx.ucsc.edu

Submitted-by: uunet!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 1, Issue 64
Archive-name: poplib

[This package allows you to add subroutine libraries that are external
 to the calling program, and independent of that program's language.
 Examples are provided; the author did not wish to release the source
 to his popup library, which is being posted to the binaries group. -sg]

#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
```

```
#
#
#
#    Run the following text with /bin/sh to create:
#      README
#      README.POP
#      MAKEFILE
#      DEBUG.H
#      LIBS.C
#      LIBS.H
#      POPTEST.C
#      POPUP.H
#      XLIB.C
#      XLIB.H
#      XTEST.C
#
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
sed 's/^X//' << 'SHAR_EOF' > README &&
XShared Library Support - Experimental
X-------------------------------------
X
XThe included files make up the pieces you need to create and use
Xshared libraries on the Atari ST.  Note that this is a experimental
Xversion.  I would like your feedback, especially constructive
Xcomments! :-)
X
XShared libraries are a useful way of factoring the common code out of
Xyour programs.  If you have a number of programs that use many of the
Xsame routines then you may want to use a shared library.
X
XIn this scheme, shared libraries are installed at boot time.  Each
Xlibrary maintains an array of pointers to it's externally accessible
Xentry points.  Programs that use a shared library acquire a pointer to
Xthe library's entry points.  An entry point is called by jumping
Xthrough the array.
X
XThe program LIBS.PRG (containing the basic shared library support)
Xshould be placed in your AUTO folder, followed by the shared libraries
Xthemselves.  An example library XLIB.PRG is included in this
Xdistribution.  At boot time GEM will execute LIBS.PRG, installing the
Xneeded shared libraries support.  As each library is executed it will
Xperform any needed initialization and install itself in memory.
X
XNote that the order of the files in the AUTO folder is critical.  The
Xprogram LIBS.PRG must preceed any libraries.  Also note that if a
Xlibrary is dependent on the presence of another library, then the
Xlibrary on which it depends must be installed first.
X
XOne of the interested possibilities offered by this scheme is the
Xability to easily make incremental changes to existing libraries to
```

```
Xwhich you may not have the source.  If there is a known bug in a
Xlibrary, you could come up with a 'patch' that replaces the offending
Xentry in the library's jump table with a pointer to your own routine.
X
XSupport routines
X----------------
X
X-- int InitLibSupport()
X
XCalled by any program using shared libraries to initialize the shared
Xlibrary support.  Returns 0 if the shared library support is NOT
Xinstalled.
X
X-- InstallLibrary(lib_link,keep)
X
XCalled by a library to install itself.
X
X-- Library FindLibrary(lib_name,lib_version)
X
XCalled by a program to find an installed library.  Returns 0 if the
Xlibrary is not found, or a pointer to the library's array of entry
Xpoints.
X
XThis distribution contains the following:
X
X    LIBS.H        - included by all code using shared libraries
X    LIBS.C        - terminate & stay resident library support
X    LIB_INIT.C    - module linked in by programs using libraries
X    LIB_INST.C    - module linked in by installable libraries
X
X    XLIB.H        - interface to example library
X    XLIB.C        - example library
X
X    XTEST.C       - program that uses example library
X
X    MAKEFILE  - to put all of the above together
X
XThis code was developed using Mark Williams C.
SHAR_EOF
chmod 0600 README || echo "restore of README fails"
sed 's/^X//' << 'SHAR_EOF' > README.POP &&
XWhat is in this package ?
X
XThis posting contains a shared library for creating popup menus in all Atari-
Xprograms (not only GEM-Applications, but no accessories). It is based on the
Xshared library framework of Preston L. Bannister. He had an excellant idea.
XPlease read the original readme file from his distribution.
XThere is no easy way to use AES-Calls in a library started at boot-time. So i
Xchoosed the accessory form for this library. The popup.acc does not provide a
Xmenu-entry. It can only be called from inside of other programs.
X
X
XWhat is a popup menu ?
```

X
XIt is like an usual GEM-menu in the menu-bar except that it pops up wherever
Xthe mouse cursor is located when you call it. You can then select in the usual
Xway by clicking with the left mouse button when the desired item is highlighted.
XSubmenus are indicated by an arrow on the left side of the entry-text. If you
Xselect a submenu, it pops up at the current cursor location and the process of
Xselection is repeated. You can also cancel the actual menu level by clicking
Xthe right mouse button. If you press the Help-key you'll get some help on the
Xcurrent entry (or popup menus in general, if no entry is selected).
X
X
XHow can i use these menus ?
X
XYou can use them inside your own programs. Perhaps you should first look at the
Xexample source poptest.c . Normally you create a popup-menu by calling the
Xfunction DefineMenu("<menu-text>"), where <menu-text> is a list of entries, all
Xseperated by the pipe-symbol `|'. For more advanced formats you should look at
Xpopup.h, the interface documentation for popup.acc . As return value of the
Xfunction DoMenu(), which does the entire dialog, you get the number of the
Xselected entry.
X
X
XHow can i provide my own (help) functions ?
X
XThe format-specification in DefineMenu() is used like the C-function printf().
XTo specify a function to be called, when an entry is selected (the Help-key is
Xpressed), you can use the format %f (%h) and provide the function-address as
Xparameter to DefineMenu(). An example:
X
X```
X   beep()
X   {
X   putchar(7);
X   }
X
X   menu_selection()
X   {
X   menu my_menu;
X
X   my_menu = DefineMenu(" hello world ! %f",beep);
X   DoMenu(my_menu);
X   FreeMenu(my_menu);
X   }
```
X
XThe function is called with three parameters:
X    - the current GEM-handle of the virtual workstation of popup.acc
X    - the default return value of this item
X    - the string-entry of the current item
XIf you don't use AES-calls anywhere else in your program, you can use the
XFormAlert()-function to display GEM-alerts on the screen (with the usual
Xparameters of form_alert()). So you need not to link the whole GEM-stuff to
Xyour program.
X
XAny comments and suggestions are always welcome. Please mail to one of the
```

```
Xaddresses given in the popup.h file or when getting the generic-help message.
X
XEdgar Roeder
SHAR_EOF
chmod 0600 README.POP || echo "restore of README.POP fails"
sed 's/^X//' << 'SHAR_EOF' > MAKEFILE &&
X# makefile for loadable library code
X
XDEBUG   = -DDEBUGGING
XCFLAGS  = $(DEBUG)
XCOMPILE = $(CC) $(CFLAGS)
X
Xall : libs.prg xlib.prg xtest.prg poptest.prg
X
X# Shared library support code - installed at boot time
Xlibs.prg : libs.c
X    $(COMPILE) libs.c
X
X# Example shared library
Xxlib.prg : xlib.o lib_inst.o
X    $(COMPILE) xlib.o lib_inst.o
X
X# Example program that uses the example shared library
Xxtest.prg : xtest.o lib_init.o
X    $(COMPILE) xtest.o lib_init.o
X
X# Example program that uses the popup menu shared library
Xpoptest.prg : poptest.o lib_init.o
X    $(COMPILE) poptest.o lib_init.o
X
Xlibs.prg xlib.o xtest.o lib_inst.o lib_init.o : libs.h
X
X# Note!  The order of installation is very important!
Xinstall  : libs.prg xlib.prg
X    -rm c:\auto\libs.prg c:\auto\xlib.prg
X    cp libs.prg xlib.prg c:\auto
X
XSRC = \include\debug.h libs.h libs.c xlib.h xlib.c xtest.c popup.h poptest.c
X
Xshar    :
X    shar readme makefile $(SRC) > libs.sh
X
XBIN = libs.prg xlib.prg xtest.prg popup.acc poptest.prg
X
Xarc : $(BIN)
X    arc -u libs.arc readme makefile $(SRC) $(BIN)
X
Xclean   :
X    $(RM) *.o *.prg libs.arc libs.sh
SHAR_EOF
chmod 0600 MAKEFILE || echo "restore of MAKEFILE fails"
sed 's/^X//' << 'SHAR_EOF' > DEBUG.H &&
X#ifdef DEBUGGING
```

```
X#define DEBUG(x)  printf x
X#define ENTER(x)  printf("entering: %s\n","x")
X#define LEAVE(x)  printf("leaving: %s\n","x")
X#else
X#define DEBUG(x)
X#define ENTER(x)
X#define LEAVE(x)
X#endif
SHAR_EOF
chmod 0600 DEBUG.H || echo "restore of DEBUG.H fails"
sed 's/^X//' << 'SHAR_EOF' > LIBS.C &&
X/*
X    Shared library support.
X    written October, 1987
X    by Preston L. Bannister
X*/
X
X#include <osbind.h>
X#include <bios.h>
X#include <basepage.h>
X#include <debug.h>
X#define LIBSUPPORT_IMPLEMENTATION
X#include "libs.h"
X
XLibLink *Libraries;
X
X
X/*===== entry points =====*/
X
X
X/*
X    Add a new library to the list of installed libraries.
X*/
X
Xvoid
XAddLibrary (lib)
X  LibLink *lib;       /* IN a new library link */
X{
X  ENTER(AddLibrary);
X  lib->next = Libraries;
X  Libraries = lib;
X  LEAVE(AddLibrary);
X}
X
X
X/*
X    Find an installed library to match the given name and version.
X    Returns a NULL pointer if the library was not found or if the
X    entrance routine of the library itself decided it was not
X    compatible with the given version.
X*/
X
XLibrary
```

```
XFindLibrary (name,version)
X  char *name;          /* IN name of the library to find */
X  int version;         /* IN version number of library */
X{
X  LibLink *p = Libraries;
X  ENTER(FindLibrary);
X  DEBUG(("Looking for '%s'\n",name));
X  for (p=Libraries; p; p=p->next)
X  {
X    DEBUG(("Examining '%s'\n",p->name));
X    if (strcmp(name,p->name)==0)
X    {
X      LEAVE(FindLibrary:FoundIt);
X      return ((*p->entrance)(version));
X    }
X  }
X  LEAVE(FindLibrary:NotFound);
X  return 0L;
X}
X
X
X/*
X   For each installed library, call the given procedure with a pointer
X   to the given data and a pointer to the library link structure.
X*/
X
Xvoid
XForEachLibrary (proc,data)
X  int (*proc)();
X  int *data;
X{
X  LibLink *p = Libraries;
X  ENTER(ForEachLibrary);
X  for (p=Libraries; p; p=p->next)
X  {
X    if ((*proc)(data,p))
X      break;
X  }
X  LEAVE(ForEachLibrary);
X}
X
X
X/*===== =====*/
X
X
XProcedure LibSupportBase[] = {
X  AddLibrary,
X  FindLibrary,
X  ForEachLibrary
X};
X
X
X/* Called when client finds this library */
```

```
X
XLibrary
XEntrance (version)
X  int version;
X{
X  DEBUG(("In LibSupport Entrance\n"));
X  switch (version)
X  {
X    case LIBSUPPORT_VERSION:
X      return LibSupportBase;
X  }
X  return 0L;
X}
X
X
XLibLink LibSupport = {
X  0L,
X  LIBSUPPORT_NAME,
X  LIBSUPPORT_VERSION,
X  Entrance
X};
X
X
Xmain ()
X{
X  extern int end[];
X
X  DEBUG(("\nInstalling LibSupport\n"));
X  Libraries = &LibSupport;
X  Setexc(LIB_FIND_VECTOR,FindLibrary);
X  DEBUG(("LibSupport vector 0x%04x set to 0x%08X FindLibrary\n",
X          LIB_FIND_VECTOR,FindLibrary));
X  Setexc(LIB_ADD_VECTOR,AddLibrary);
X  DEBUG(("LibSupport vector 0x%04x set to 0x%08X AddLibrary\n",
X          LIB_ADD_VECTOR,AddLibrary));
X  Ptermres((long)end-(long)BP,0);
X  DEBUG(("LibSupport installation failed!\n"));
X  return 1;
X}
SHAR_EOF
chmod 0600 LIBS.C || echo "restore of LIBS.C fails"
sed 's/^X//' << 'SHAR_EOF' > LIBS.H &&
X/*
X   Definitions for shared library support
X   written November, 1987
X   by Preston L. Bannister
X*/
X
X#define LIBSUPPORT_NAME         "LibSupport"
X#define LIBSUPPORT_VERSION 1
X
X#define LIB_FIND_VECTOR    0x200
X#define LIB_ADD_VECTOR 0x201
```

```
X
Xtypedef long (*Procedure)();
Xtypedef Procedure * Library;
Xtypedef Library (*GetLibraryFunction)();
X
Xtypedef struct _LibLink {
X  struct _LibLink *next;
X  char *name;
X  int version;
X  GetLibraryFunction entrance;
X} LibLink;
X
Xextern Library LibSupportBase;
X
Xextern void InstallLibrary(/*lib,keep*/);
Xextern int  InitLibSupport();
X
X#ifndef LIBSUPPORT_IMPLEMENTATION
X
X#define AddLibrary    (*LibSupportBase[0])
X#define FindLibrary   (*LibSupportBase[1])
X#define ForEachLibrary(*LibSupportBase[2])
X
X#endif
SHAR_EOF
chmod 0600 LIBS.H || echo "restore of LIBS.H fails"
sed 's/^X//' << 'SHAR_EOF' > POPTEST.C &&
X#include <osbind.h>
X#include "libs.h"
X#include "popup.h"
X
XLibrary PopupLibBase; /* points to popup menu library */
X
XValue
Xbeep(handle,i)
X{
X    Bconout(2,7);
X    return(42);
X}
X
Xhelp(handle,i,str)
Xchar    *str;
X{
X    char out[256];
X
X    sprintf(out,"[0][ If you select | '%s' | you'll hear the bell ][ OK ]",
X         str);
X    FormAlert(1,out);
X}
X
Xmain ()
X{
X    menu test_menu;
```

```
X    menu sub_menu;
X    Value result;
X    char out[80];
X    char out_str[256];
X
X    if (InitLibSupport()) {
X         if (PopupLibBase = (Library)
X                   FindLibrary(POPUP_LIB,POPUP_VERSION)) {
X              sprintf(out_str,"[3][ Library | '%s' ][ 0x%08X ]",
X                   POPUP_LIB,PopupLibBase);
X              FormAlert(1,out_str);
X              ConfigRows(2);
X              sub_menu =
X              DefineMenu(" Submenu %s%t| five %v| Beep %h%f",
X                   5,help,beep);
X              test_menu =
X              DefineMenu(" test %s%t| one | two | Submenu %o%m",
X                   sub_menu);
X              result = DoMenu(test_menu);
X              out[0] = '\0';
X              switch(result) {
X                   case -2 : strcat(out,"cancelled");
X                   case -1 : strcat(out,"| no selection"); break;
X                   default : sprintf(out,"you selected %d",result);
X              }
X              sprintf(out_str,"[0][ %s ][ OK ]",out);
X              FormAlert(1,out_str);
X              FreeMenu(test_menu);
X              FreeMenu(sub_menu);
X              exit(0);
X         } else {
X              printf("PopupLib not available\n");
X         }
X    } else printf("no library support\n");
X    exit(1);
X}
SHAR_EOF
chmod 0600 POPTEST.C || echo "restore of POPTEST.C fails"
sed 's/^X//' << 'SHAR_EOF' > POPUP.H &&
X/*************************************************************************/
X/* @(#) popup.h     1.0 -- standard include file for popup menu usage 09.07.88
*/
X/*                                                 */
X/* This header file and the corresponding shared library are in the public    */
X/* domain. They can be copied and distributed for non-commercial usage.       */
X/*                                                 */
X/* (c) 10/1987 by Edgar Roeder                              */
X/*           Liesbet-Dill-Str. 3                            */
X/*           D-6602 Dudweiler                          */
X/*           W-Germany                              */
X/*           E-mail: roeder@sbsvax.UUCP or                    */
X/*                roeder@eansb.informatik.uni-saarland.dbp&de   */
X/*************************************************************************/
```

```
X
X#define POPUP_LIB "popup menus"
X#define POPUP_VERSION 1
X
X/*
X * usage of this package:
X *
X * menu_id = DefineMenu(str,args);
X * value = DoMenu(menu_id);
X * err = FreeMenu(menu_id);
X *
X * example:
X *
X * long  menu_id;
X * menu_id = DefineMenu(" testmenu %t| first | second | third ");
X * switch(DoMenu(menu_id)) {
X *  case -2 : printf("Right mouse cancelled menu\n"); break;
X *  case -1 : printf("No menu item selected\n"); break;
X * ...
X * }
X * if(FreeMenu(menu_id)) fprintf(stderr,"Cannot free this menu");
X *
X */
X
Xtypedef long menu;
Xtypedef int  Value;
X
X/*****************************************************************************
X *                                                                           *
X * DefineMenu   returns a new menu id, which can be used to display the      *
X *              defined menu or 0 if an error occured                        *
X * DoMenu       returns as standard value the number of the selected item    *
X *              -2 is returned if the button is pressed outside the menu     *
X *              -1 is returned if no selection was made                      *
X *               0 is returned if there was an error                         *
X * FreeMenu     returns 0 if no error occured (menu_id is deleted)           *
X * ConfigRows   sets the maximum number of rows in a column (default is 5)   *
X *              and returns the old value for this                           *
X * ConfigScreen sets the border of the screen usable to display popup menus  *
X *              (the default values are: 0, 0, 639, 399 for mono screen)      *
X *                                                                           *
X *    complete definition of advanced menu formats (decreasing priority):    *
X *                                                                           *
X * format modifier |  required arguments  |          effect                 *
X * ----------------+----------------------+---------------------------------- *
X *       %t        | no additional args   | make a non selectable menu title  *
X * ----------------+----------------------+---------------------------------- *
X *       %v        | one int argument     | supply non standard return value  *
X *                 |                      |                                   *
X * ----------------+----------------------+---------------------------------- *
X *       %m        | one menu identifier  | invoke submenu                    *
X *                 |                      |                                   *
X * ----------------+----------------------+---------------------------------- *
```

```
X *       %c        | no additional args   | continue with menu if this item   *
X *                 |                      | is selected or no selection       *
X * ----------------+----------------------+---------------------------------- *
X *       %x        | no additional args   | continue with menu if no entry    *
X *                 |                      | selected                          *
X * ----------------+----------------------+---------------------------------- *
X *       %s        | no additional args   | selection required in this part   *
X * ----------------+----------------------+---------------------------------- *
X *       %o        | no additional args   | overlay this submenu (or all)     *
X * ----------------+----------------------+---------------------------------- *
X *       %f        | one function pointer | call function with standard       *
X *                 |                      | value as single argument          *
X * ----------------+----------------------+---------------------------------- *
X *       %h        | one function pointer | call function if user pressed     *
X *                 |                      | the Help key                      *
X *                                                                             *
X * if more than one format is given, the effects are occuring in order of     *
X * decreasing priority (you could attach a function to a whole menu with      *
X * %f and %m given in any order, or invoke the function with a non standard   *
X * value given with %v %f, in %m %v format the non standard value overrides   *
X * a return value of -1 if no selection occured during submenu invocation)    *
X *                                                                             *
X *****************************************************************************/
X
X#ifdef LATTICE
Xextern Procedure *PopupLibBase;
X#else
Xextern Library PopupLibBase;
X#endif LATTICE
X
X#ifdef POPUP_IMPLEMENTATION
Xextern menu  DefineMenu(); /* define a popup menu                        */
Xextern long  FreeMenu();   /* free the given menu                        */
Xextern Value DoMenu();/* display the given menu and handle input    */
Xextern int   ConfigMenu(); /* configure display parameters for defined   */
X                 /* menus                      */
X
X/* this function is used by DoMenu with the actual mouse coordinates as input */
Xextern Value DoXYMenu();   /* display menu at given coordinates          */
X                 /* usage: DoXYMenu(screen_handle,menu_id,x,y) */
X#else
X#define DefineMenu         (*PopupLibBase[0])
X#define DoMenu             (*PopupLibBase[1])
X#define FreeMenu       (*PopupLibBase[2])
X#define DoXYMenu       (*PopupLibBase[3])
X#define ConfigMenu         (*PopupLibBase[4])
X#define ConfigRows(r)      ConfigMenu(r)
X#define ConfigScreen(x,y,w,h)   ConfigMenu(0,x,y,w,h)
X#define FormAlert      (*PopupLibBase[5])
X#endif
SHAR_EOF
chmod 0600 POPUP.H || echo "restore of POPUP.H fails"
sed 's/^X//' << 'SHAR_EOF' > XLIB.C &&
```

```
X/* Test Library Implementation */
X
X#include "libs.h"
X#define XLIB_IMPLEMENTATION
X#include "xlib.h"
X#include <debug.h>
X
X
X/*===== Library Entry Points =====*/
X
X
Xvoid XLibP0 () { DEBUG(("in XLibP0\n")); }
Xvoid XLibP1 () { DEBUG(("in XLibP1\n")); }
Xvoid XLibP2 () { DEBUG(("in XLibP2\n")); }
X
X
X/*===== Overhead =====*/
X
X
X/* array of entry points for this library */
X
XProcedure XLibBase[] = {
X  XLibP0,
X  XLibP1,
X  XLibP2
X};
X
X
X/*
X    This routine is called by FindLibrary when a program wants
X    to use the library.  If the version number given by the caller
X    is not acceptable we will return NULL.  If we wanted to support
X    more than one version with the same library, we could choose to
X    return a different library base pointer for some versions.
X*/
X
XLibrary
XEntrance (version)
X  int version;
X{
X  DEBUG(("XLib Entrance with version %d\n",version));
X  if (version==XLIB_VERSION)
X    return XLibBase;
X  return 0L;
X}
X
X
X/* The link for this library */
X
XLibLink XLib = {
X  0L,
X  XLIB_NAME,
X  XLIB_VERSION,
```

```
X  Entrance
X};
X
X
X/* initialization code for library */
X
Xint
Xmain ()
X{
X  /* install this library */
X  return InstallLibrary(&XLib,0L);
X}
SHAR_EOF
chmod 0600 XLIB.C || echo "restore of XLIB.C fails"
sed 's/^X//' << 'SHAR_EOF' > XLIB.H &&
X/* Example Library Interface */
X
X#define XLIB_NAME "XLib"
X#define XLIB_VERSION  1
X
Xextern Library XLibBase;
X
X#ifndef XLIB_IMPLEMENTATION
X
X#define XLibP0    (*XLibBase[0])
X#define XLibP1    (*XLibBase[1])
X#define XLibP2    (*XLibBase[2])
X
X#endif
SHAR_EOF
chmod 0600 XLIB.H || echo "restore of XLIB.H fails"
sed 's/^X//' << 'SHAR_EOF' > XTEST.C &&
X/*
X    Shared library test
X    written November, 1987
X    by Preston L. Bannister
X*/
X
X#include <debug.h>
X#include "libs.h"
X#include "xlib.h"
X
XLibrary XLibBase; /* points to our example library */
X
Xmain ()
X{
X  if (InitLibSupport())
X  {
X    printf("Attempting to find library '%s'\n",XLIB_NAME);
X    if (XLibBase = (Library)FindLibrary(XLIB_NAME,XLIB_VERSION))
X    {
X      printf("XLibBase 0x%08X\n",XLibBase);
X      DEBUG(("Calling XLibP0 @ 0x%08X\n",XLibBase[0]));
```

```
X      XLibP0();
X      DEBUG(("Calling XLibP1 @ 0x%08X\n",XLibBase[1]));
X      XLibP1();
X      DEBUG(("Calling XLibP2 @ 0x%08X\n",XLibBase[2]));
X      (void)XLibP2();
X    }
X    else
X    {
X      printf("XLib not availible??\n");
X    }
X  }
X  exit(0);
X}
SHAR_EOF
chmod 0600 XTEST.C || echo "restore of XTEST.C fails"
exit 0


From -7550348559824354864
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,7c428cf5cebf3647,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-07-29 10:06:53 PST
Path: utzoo!utgpu!water!watmath!clyde!att!osu-cis!tut.cis.ohio-state.edu!husc6!
bbn!uwmcsd1!ig!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i065: uupc -- UUCP clone (pre-pre-release)  part01/04
Keywords: Laser, buggy
Message-ID: <4332@saturn.ucsc.edu>
Date: 29 Jul 88 17:06:53 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1969
Approved: koreth@ssyx.ucsc.edu

Submitted-by: koreth@ssyx.ucsc.edu (Steven Grimm)
Posting-number: Volume 1, Issue 65
Archive-name: uupc/part01

This is the version of UUPC that I've been working on.  It allows you
to use your ST as a UUCP node.  Please look at the file "readme.1st"
before proceeding; it contains configuration instructions and some
caveats about the code.  It's about halfway through part03 if you
don't want to unshar before reading it.

#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
# This is part 1 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
#
#    Run the following text with /bin/sh to create:
#      chekname.c
```

```
#	dcp.c
#	dcp.h
#	dcpgpkt.c
#	dcpsys.c
#	dcpxfer.c
#	dirsubs.c
#	dmymath.c
#	fgets.c
#	genv.c
#	genv.h
#	getargs.c
#	getenv.c
#	getone.c
#	host.c
#	host.h
#	inoutpth.c
#	lib.c
#	lmail.c
#	mail.c
#	mailhost.c
#	makefile
#	mlib.c
#	ndir.c
#	ndir.h
#	pcmail.c
#	printmsg.c
#	readme.1st
#	readme.dcp
#	readme.inf
#	readme.por
#	readme.st
#	readme.uup
#	rmail.c
#	shell.c
#	systems
#	time.c
#	time.h
#	ulib.c
#	uuhost.c
#	uusup.c
#
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
sed 's/^X//' << 'SHAR_EOF' > chekname.c &&
X#include "dcp.h"
X#include <ctype.h>
X
Xextern char s_systems[];
XFILE *FOPEN();
X
X/*
```

```
X**
X**checkname
X** Do we know the guy ?
X*/
Xcheckname(name)
Xchar    name[];
X{
X    FILE *ff;
X    char line[BUFSIZ], tmp[20]; /* can change to 8 if %8s works */
X
X    if ( ( ff = FOPEN( s_systems, "r", 't' )) == (char *)NULL )
X        return( FAILED );
X
X    while ( fgets( line, BUFSIZ, ff ) ){
X        sscanf( line, "%8s ", tmp );
X        if ( strncmp( tmp, name, 7 ) == 0 ) {
X            fclose( ff );
X            return ( OK ); /*OK I like you */
X        }
X    }
X    fclose( ff );
X    return( FAILED ); /* Who are you ? */
X
X}
X
X/*
X * Non case-sensitive string compare
X */
Xstrncom(s1, s2, n)
Xregister char *s1, *s2;
X{
X    register int t1, t2;
X
X    while ((*s1 || *s2) && n--)
X        if ((t1 = toupper(*s1++)) < (t2 = toupper(*s2++)))
X            return -1;
X        else if (t1 > t2)
X            return 1;
X
X    return 0;
X}
SHAR_EOF
chmod 0600 chekname.c || echo "restore of chekname.c fails"
sed 's/^X//' << 'SHAR_EOF' > dcp.c &&
X/*          dcp.c
X
X            Revised edition of dcp
X
X            Stuart Lynne May/87
X
X            Copyright (c) Richard H. Lamb 1985, 1986, 1987
X            Changes Copyright (c) Stuart Lynne 1987
X
```

```
X*/
X/* "DCP" a uucp clone. Copyright Richard H. Lamb 1985,1986,1987 */
X/* This program implements a uucico type file transfer and remote
Xexecution type protocol.
X*/
X#include "dcp.h"
X/**/
Xint pktsize;                    /* packet size for pro */
XFILE    *logfile;              /* system log file */
XFILE    *syslog;               /* system log file */
XFILE    *fw;                   /* cfile pointer */
Xchar    state;                      /* system state*/
Xchar    cfile[80];              /* work file pointer */
Xint     remote;                 /* -1 means we're remote ..7*/
Xint     msgtime;                /* timout setting */
Xchar    fromfile[132];
Xchar    hostfile[132];         /* host version of fromfile */
Xchar    tofile[132];
Xint     fp;                     /* current disk file ptr */
Xint     size;                   /* nbytes in buff */
XFILE    *fsys;
Xchar    Rmtname[20];
Xchar    rmtname[20];
Xchar    * cctime;
Xchar    proto[5];
X/* char  loginseq[256]; */
Xchar    sysline[BUFSIZ];
Xchar    s_systems[64];
Xchar    s_logfile[64];
Xchar    s_syslog[64];
Xchar    *flds[60];
Xint     kflds;
Xint     debuglevel;        /* debugging flag */
X
Xunsigned int checksum();
X
X
X/**/
X/* new usage
X
X    dcp  [-xn] -r0     slave mode
X    dcp  [-xn] -shost       call host
X    dcp  [-xn] -sall        call all hosts
X
X    dcp  [-xn]         call any hosts as required by C. files
X
X*/
X
Xcomplain( s )
Xchar *s;
X{
X    fprintf( stderr, "Please set your %s environment variable.", s );
X}
```

```
X
Xstatic void cant( file )
Xchar *file;
X{
X    fprintf( stderr, "Can't open: \"%s\"\n", file );
X    exit( NULL );
X}
X
Xdcpmain( argc, argv )
Xint argc;
Xchar*argv[];
X{
X    FILE *ftmp;
X    char line[132];
X
X    if ( name == NULL || *name == '\0' ) {
X         complain( NAME );
X         exit( -1 );
X    }
X    if ( nodename == NULL || *nodename == '\0' ) {
X         complain( NODENAME );
X         exit( -1 );
X    }
X
X    mkfilename( s_logfile, spooldir, LOGFILE );
X    mkfilename( s_syslog,  spooldir, SYSLOG  );
X    mkfilename( s_systems, confdir,  SYSTEMS );
X
X    if ( (logfile = FOPEN( s_logfile, "a", 't' )) == NULL )
X       cant( s_logfile );
X    if ( (syslog  = FOPEN( s_syslog,  "a", 't' )) == NULL )
X       cant( s_syslog );
X
X    remote = MASTER;
X    debuglevel = 1;
X    fp = -1;
X    fw = (char *)NULL;
X
X
X    remote = MASTER;
X    strcpy( Rmtname, "any" );
X
X    while ( --argc ) {
X         if ( **++argv == '-') {
X              switch(argv[0][1]) {
X              case 'x':
X                   debuglevel = atoi( &argv[0][2] );
X                   break;
X              case 's':
X                   sprintf( Rmtname, "%.7s", &argv[0][2] );
X                   break;
X              case 'r':
X                   remote = atoi( &argv[0][2] );
```

```
X                    break;
X
X               default:
X                    break;
X               }
X          }
X     }
X
X
X     if ( remote == MASTER ) {
X          printmsg( 0, "Calling %s, debuglevel=%d", Rmtname, debuglevel );
X          if (( fsys = FOPEN( s_systems, "r", 't' )) == (char *)NULL )
X               exit( FAILED );
X          state = 'I';
X
X          while (TRUE) {
X               printmsg( 4, "Mstate = %c", state );
X               switch (state) {
X               case 'I':
X                    state = getsystem();
X                    break;
X               case 'S':
X                    state = callup();
X                    break;
X               case 'P':
X                    state = startup();
X                    break;
X               case 'D':
X                    state = master();
X                    break;
X               case 'Y':
X                    state = sysend();
X                    break;
X               case 'G':
X                    state = 'I';
X                    break;
X               }
X               if (state == 'A')
X                    break;
X          }
X          fclose( fsys );
X     } else
X      {
X          if (openline( device, speed ) == -1)
X               return(FALSE);
X          state = 'L';
X          while (TRUE) {
X               printmsg( 4, "Sstate = %c", state );
X               switch (state) {
X               case 'L':
X                    state = login();
X                    break;
X               case 'I':
```

```
X                 state = startup();
X                 break;
X            case 'R':
X                 state = slave();
X                 break;
X            case 'Y':
X                 state = sysend();
X                 break;
X            }
X            if (state == 'A')
X                 break;
X        }
X        closeline();
X    }
X
X    printmsg( 2, "calling dcxqt" );
X    if (dcxqt())
X        printmsg( 0, "ERROR in DCXQT" );
X
X    /* scan and process any recieved files */
X
X    fclose( syslog );
X    fclose( logfile );
X    return 0;
X}
X
X
X/**/
X/*
X**
X**
X**master
X**
X**
X*/
Xmaster()
X{
X    state = 'I';
X    while (TRUE) {
X        printmsg( 4, "Top level state (master mode) %c", state );
X        switch (state) {
X        case 'I':
X            state = sinit();
X            break;
X        case 'B':
X            state = scandir();
X            break;
X        case 'S':
X            state = send();
X            break;
X        case 'Q':
X            state = sbreak();
X            break;
```

```
X        case 'G':
X             state = receive();
X             break;
X        case 'C':
X             state = 'Y';
X             break;
X        case 'Y':
X             state = endp();
X             break;
X        case 'P':
X             return('Y');
X        case 'A':
X             return('A');
X        default:
X             return('A');
X        }
X    }
X}
X
X
X/**/
X/*
X**
X**
X**slave
X**
X**
X*/
Xslave()
X{
X    state = 'I';
X    while (TRUE) {
X         printmsg( 4, "Top level state (slave mode) %c", state );
X         switch (state) {
X         case 'I':
X              state = rinit();
X              break;
X         case 'F':
X              state = receive();
X              break;
X         case 'C':
X              state = schkdir();
X              break;
X         case 'T':
X              state = 'B';
X              break;
X         case 'B':
X              state = scandir();
X              break;
X         case 'S':
X              state = send();
X              break;
X         case 'Q':
```

```
X             state = sbreak();
X             break;
X        case 'G':
X             return('Y');
X        case 'Y':
X             state = endp();
X             break;
X        case 'P':
X             return('Y');
X        case 'A':
X             return('A');
X        default:
X             return('A');
X        }
X    }
X}
X
X
X/**/
X/*
X *  r e c e i v e
X *
X *  This is the state table switcher for receiving files.
X */
X
Xreceive()
X{
X
X    state = 'F';/* Receive-Init is the start state */
X
X    while (TRUE) {
X         printmsg( 4, " receive state: %c", state );
X         switch (state)/* Do until done */ {
X         case 'F':
X              state = rfile();
X              break; /* Receive-File */
X         case 'D':
X              state = rdata();
X              break; /* Receive-Data */
X         case 'C':
X              return('C');/* Complete state */
X         case 'A':
X              return('Y');/* "Abort" state */
X         default:
X              return('Y');
X         }
X    }
X}
X
X
X/**/
X/*
X *  s e n d
```

```
X *
X *  Sendsw is the state table switcher for sending files.  It loops until
X *  either it finishes, or an error is encountered.  The routines called
X *  by sendsw are responsible for changing the state.
X *
X */
Xsend()
X{
X    fp = -1;                /* reset file getter/opener */
X    state = 'F';/* Send initiate is the start state */
X    while (TRUE)/* Do this as long as necessary */ {
X         printmsg( 4, "send state: %c", state );
X         switch (state) {
X         case 'F':
X              state = sfile();
X              break; /* Send-File */
X         case 'D':
X              state = sdata();
X              break; /* Send-Data */
X         case 'Z':
X              state = seof();
X              break; /* Send-End-of-File */
X         case 'B':
X              return ('B'); /* Complete */
X         case 'A':
X              return ('Y'); /* "Abort" */
X         default:
X              return ('Y'); /* Unknown, fail */
X         }
X    }
X}
X
X
X/**/
X/* A command formatter for DCP. RH Lamb */
X/* sets up stdin and stdout on various machines */
X/* There is NO command checking so watch what you send and who you */
X/* let accsess your machine. "C rm /usr/*.*" could be executed. */
Xdcxqt()
X{
X    int  i;
X    char command[60], input[60], output[60], line[BUFSIZ];
X    char *cp;
X
X    rmtname[0] = '\0';
X
X    while (dscandir()) {
X         strcpy( line, cfile );
X         fw = FOPEN( line, "r", 'b' );/* imported X file */
X         strcpy(cfile, line);
X         printmsg( 2, "dcxqt:%s %ld", cfile, fw );
X         input[0]   = '\0';
X         output[0]  = '\0';
```

```
X         command[0] = '\0';
X         while ( fgets( line, BUFSIZ, fw ) ) {
X
X              cp = index( line, '\n' );
X              if ( cp != (char *)NULL )
X                   *cp = '\0';
X
X              printmsg( 8, "dcxqt: %s", line );
X              switch (line[0]) {
X              case 'U':
X                   break;
X              case 'I':
X                   strcpy( input,   &line[2] );
X                   break;
X              case 'O':
X                   strcpy( output,  &line[2] );
X                   break;
X              case 'C':
X                   strcpy( command, &line[2] );
X                   break;
X              case 'R':
X                   break;
X              default :
X                   break;
X              }
X         }
X         fclose( fw );
X
X         printmsg( 0, "xqt: %s\n", command );
X
X         shell( command, input, output, (char *)NULL );
X
X
X         unlink(cfile);
X         importpath( hostfile, input );
X         unlink(hostfile);
X         importpath( hostfile, output );
X         unlink(hostfile);
X    }
X    return(0);
X}
X
X/**/
X/*
X *  p r i n t m s g
X *
X *  Print error message on standard output if not remote.
X */
X/*VARARGS1*/
Xprintmsg(level, fmt, a1, a2, a3, a4, a5)
Xint      level;
Xchar     *fmt;
Xchar     *a1, *a2, *a3, *a4, *a5;
```

```
X{
X    char msg[256];
X
X    if ( debuglevel > level ) {
X         sprintf( msg, fmt, a1, a2, a3, a4, a5 );
X         strcat( msg, "\n" );
X         if ( remote == MASTER )
X              fputs( msg, stdout );
X         fputs( msg, logfile );
X    }
X}
X
X
X
X
SHAR_EOF
chmod 0600 dcp.c || echo "restore of dcp.c fails"
sed 's/^X//' << 'SHAR_EOF' > dcp.h &&
X/* DCP a uucp clone. Copyright Richard H. Lamb 1985,1986,1987 */
X#include <stdio.h>         /* Standard UNIX  definitions */
X#include "host.h"     /* Host specific definitions */
X
XFILE * FOPEN();
Xint CREAT();
X
X#define SYSTEMS       "systems"
X#define LOGFILE       "LOGFILE"
X#define SYSLOG        "SYSLOG"
X
X#define MSGTIME         20
X#define MAXPACK         256
X
X#define ACK     4       /*      general definitions     */
X#define NAK     2
X#define DATA    0
X#define CLOSE   1
X#define ERROR   10
X#define EMPTY   11
X
X
X
X#define  SLAVE    0
X#define  MASTER   1
X
X#ifndef TRUE
X#define TRUE   (-1)
X#define FALSE   0
X#endif
X
X#ifndef SAME
X#define SAME 0
X#endif
X#define FAILED    -1
```

```
X#define OK       0
X
X
X/* L.sys field defines */
X
X#define FLD_REMOTE    0
X#define FLD_CCTIME    1
X#define FLD_DEVICE    2
X#define FLD_TYPE 3
X#define FLD_SPEED 4
X#define FLD_PROTO 5
X#define FLD_EXPECT    6
X#define FLD_SEND  7
X
X/**/
Xtypedef int  (*procref)();
X
Xtypedef struct {
X    char type;
X    procref a;
X    procref b;
X    procref c;
X    procref d;
X} Proto;
X
X/* the various protocols available. Add here for others */
Xextern procref          getpkt, sendpkt, openpk, closepk;
X
Xextern int   ggetpkt(), gsendpkt(), gopenpk(), gclosepk();
X/*
Xextern int   kgetpkt(), ksendpkt(), kopenpk(), kclosepk();
Xextern int   rgetpkt(), rsendpkt(), ropenpk(), rclosepk();
Xextern int   tgetpkt(), tsendpkt(), topenpk(), tclosepk();
X*/
X
X/**/
Xextern int   pktsize;               /* packet size for this pro*/
Xextern FILE  *logfile;             /* system log file */
Xextern FILE  *syslog;              /* system log file */
Xextern FILE  *fw;                  /* cfile pointer */
Xextern char  cfile[80];             /* work file pointer */
Xextern int   remote;               /* -1 means we're remote*/
Xextern int   findwork;
Xextern int   msgtime;              /* timout setting */
Xextern char  fromfile[132];
Xextern char  hostfile[132];        /* host version of fromfile */
Xextern char  tofile[132];
Xextern char  state;                /* present state */
Xextern int   fp;                   /* current disk file ptr */
Xextern int   size;                 /* nbytes in buff */
Xextern FILE  *fsys;
Xextern char  Rmtname[20];
Xextern char  rmtname[20];
```

```
Xextern char  *cctime;
Xextern char  proto[5];
X
Xextern char sysline[BUFSIZ];
Xextern char s_systems[64];
Xextern char s_logfile[64];
Xextern char s_syslog[64];
Xextern char *flds[60];
Xextern int  kflds;
X
Xextern int   debuglevel;        /* debugging flag */
Xextern unsigned int   checksum();
Xextern char *index();
Xextern char *rindex();
X
Xextern  char *curdir;
X
X
X
SHAR_EOF
chmod 0600 dcp.h || echo "restore of dcp.h fails"
sed 's/^X//' << 'SHAR_EOF' > dcpgpkt.c &&
X/*           dcpgpkt.c
X
X             Revised edition of dcp
X
X             Stuart Lynne May/87
X
X             Copyright (c) Richard H. Lamb 1985, 1986, 1987
X             Changes Copyright (c) Stuart Lynne 1987
X
X*/
X/* "DCP" a uucp clone. Copyright Richard H. Lamb 1985,1986,1987 */
X/* 3-window "g" ptotocol */
X/* Thanks got to John Gilmore for sending me a copy of Greg Chesson's
XUUCP protocol description-- Obviously invaluable
XThanks also go to Andrew Tannenbaum for the section on Siding window
Xprotocols with a program example in his "Computer Networks" book
X*/
X#include "dcp.h"
X#define PKTSIZE         64
X#define PKTSIZ2         2
X#define HDRSIZE         6
X#define MAXTRY          4
X
X#define MAXERR  200        /* Dont want to quit in a middle of a long file*/
X#define TIMEOUT 4     /* could be longer */
X#define KPKT    1024/PKTSIZE
X#define POK      -1
X
X#define SWINDOW 3 /* fixed now, U make it variable */
X#define RWINDOW 3
X#define NBUF    8   /* always SAME as MAXSEQ ? */
```

```
X#define MAXSEQ  8
X
X#define between(a,b,c) ((a<=b && b<c) || (c<a && a<=b) || (b<c && c<a))
X
X/* packet defin */
Xstatic int   rwl, swl, swu, rwu, nerr, nbuffers, npkt, irec, timeout, GOT_SYNC,
GOT_HDR;
Xstatic int   fseq[NBUF], outlen[NBUF], inlen[NBUF], arr[NBUF];
Xstatic char  outbuf[NBUF][PKTSIZE+1], inbuf[NBUF][PKTSIZE+1];
Xstatic unsigned char  grpkt[HDRSIZE+1];
Xstatic long  ftimer[NBUF], acktmr, naktmr;
X/**/
X/******************SUB SUB SUB PACKET HANDLER************/
Xgopenpk()
X{
X    int  i, j, n1, n2, len;
X    char tmp[PKTSIZE+1];
X    pktsize = PKTSIZE; /* change it later after the init */
X    msgtime = MSGTIME; /* not sure I need this for "g" proto */
X    /* initialize proto parameters */
X    swl = nerr = nbuffers = npkt = 0;
X    swl = swu = 1;
X    rwl = 0;
X    rwu = RWINDOW - 1;
X    for (i = 0; i < NBUF; i++) {
X         ftimer[i] = 0;
X         arr[i] = FALSE;
X    }
X    GOT_SYNC = GOT_HDR = FALSE;
X    /* 3-way handshake */
X    timeout = 1; /* want some timeout capability here */
X    gspack(7, 0, 0, 0, tmp);
Xrsrt:
X    if (nerr >= MAXERR)
X         return(-1);
X    /* INIT sequence. Easy fix for variable pktsize and windows. */
X    /* I didnt since all the machines I talk to use W=3 PKTSZ=64 */
X    /* If you do this make sure to reflect the changes in "grpack" */
X    /* and "gspack" */
X    switch (grpack(&n1, &n2, &len, tmp)) {
X    case 7:
X         gspack(6, 0, 0, 0, tmp);
X         goto rsrt;
X    case 6:
X         gspack(5, 0, 0, 0, tmp);
X         goto rsrt;
X    case 5:
X         break;
X    default:
X         nerr++;
X         gspack(7, 0, 0, 0, tmp);
X         goto rsrt;
X    }
```

```
X    nerr = 0;
X    return(0);      /* channel open */
X}
X
X
Xgclosepk()
X{
X    int  i;
X    char tmp[PKTSIZE+1];
X    timeout = 1;
X    for (i = 0; i < MAXTRY; i++) {
X         gspack(CLOSE, 0, 0, 0, tmp);
X         if (gmachine() == CLOSE)
X              break;
X    }
X    printmsg( 0, "number of errors %d and pkts xfered %d", nerr, npkt );
X    return(0);
X}
X
X
X/*
X *
X * ggetpkt
X ***** description: Gets no more than a packet's worth of data from
X****               the "packet i/o state machine". May have to
X****               periodically run the pkt machine to get some
X****               packets.
X* on input: dont care   getpkt(data,&len)  char *data int len
X* on return: data+\0 and length in len. ret(0) if alls well
X* ret(-1) if problems.(fail)
X */
Xggetpkt(cdata, len)
Xint *len;
Xchar    cdata[];
X{
X    int  i2;
X    irec = 1;
X    timeout = 0;
X    /* WAIT FOR THE DESIRED PACKET */
X    while ((arr[rwl]) == FALSE)
X         if (gmachine() != POK)
X              return(-1);
X    /* GOT A PKT ! */
X    i2 = rwl; /*<-- mod(,rwindow) for larger than 8 seq no.s */
X    *len = inlen[i2];
X    strncpy(cdata, inbuf[i2], *len);
X    arr[i2] = FALSE;
X    rwu = (1 + rwu) % MAXSEQ; /* bump rec window */
X    npkt++;
X    return(0);
X}
X
X
```

```
X/*
X *
X *  sendpkt
X *
X*****   description:    Put at most a packet's worth of data  in the pkt state
X ****                   machine for xmission.
X*****                   May have to run the pkt machine a few times to get
X*****                   an available output slot.
X *
X * on input: char *data int len,flg; len=length of data in data.
X *           flg=2 just send the packet with no wait for ack.
X *           flg>0 zero out the unused part of the buffer. (for UUCP "msg"
X *                                                                  pkts)
X *           flg=0 normal data
X * return:   ret(0) if alls well ret(-1) if problems (fail)
X *
X*/
Xgsendpkt(cdata, len, flg)
Xint len, flg;
Xchar    *cdata;
X{
X    int  i, i1;
X    long ttmp;
X    irec = 0;
X    timeout = 0; /* non-blocking reads */
X    /* WAIT FOR INPUT i.e. if weve sent SWINDOW pkts and none have been */
X    /* acked, wait for acks */
X    while (nbuffers >= SWINDOW)
X         if (gmachine() != POK)
X              return(-1);
X    i1 = swu;  /* <--If we ever have more than 8 seq no.s, must mod() here*/
X    /* PLACE PACKET IN TABLE AND MARK UNACKED */
X    /* fill with zeros or not */
X    if (flg) {
X         strcpy(outbuf[i1], cdata);
X         len = PKTSIZE;
X         for (i = strlen(cdata); i < len; i++)
X              outbuf[i1][i] = '\0';
X    } else {
X         strncpy(outbuf[i1], cdata, len);
X         outbuf[i1][len] = '\0';
X    }
X    /* mark packet  */
X    outlen[i1] = len;
X    ftimer[i1] = time(&ttmp);
X    fseq[i1] = swu;
X    swu = (1 + swu) % MAXSEQ;   /* bump send window */
X    nbuffers++;
X    npkt++;
X    /* send it */
X    gspack(DATA, rwl, fseq[i1], outlen[i1], outbuf[i1]);
X    /* send it once then let the pkt machine take it. wouldnt need this for */
X    /* mtasking systems */
```

```
X    /* sl gmachine(); */
X    return(0);
X}
X
X
X/************     packet machine   ****** RH Lamb 3/87 */
X/* Idealy we would like to fork this process off in an infinite loop */
X/* and send and rec pkts thru "inbuf" and "outbuf". Cant do this in MS-DOS*/
X/* so we setup "getpkt" and "sendpkt" to call this routine often and return */
X/* only when the input buffer is empty thus "blocking" the pkt-machine task. */
Xgmachine()
X{
X    int  rack, rseq, rlen, i1, i2, dflg;
X    char rdata[PKTSIZE+1];
X    long ttmp, itmp;
Xreply:
X    printmsg( 6, "*send %d<W<%d, rec %d<W<%d, err %d", swl, swu, rwl, rwu,
nerr );
X    /* waiting for ACKs for swl to swu-1. Next pkt to send=swu */
X    /* rwl=expected pkt */
X    printmsg( 7, "Kbytes transfered %d errors %d\r", npkt / KPKT, nerr );
X    if (nerr >= MAXERR)
X         goto close;
X    dflg = 0;
X    switch (grpack(&rack, &rseq, &rlen, rdata)) {
X    case CLOSE:
X         printmsg( 5, "**got CLOSE");
X         goto close;
X    case NAK:
X         nerr++;
X         acktmr = naktmr = 0; /* stop ack/nak timer */
X         printmsg( 5, "**got NAK %d", rack );
Xnloop:
X         if (between(swl, rack, swu)) { /* resend rack->(swu-1) */
X              i1 = rack;
X              gspack(DATA, rwl, rack, outlen[i1], outbuf[i1]);
X              printmsg( 5, "***resent %d", rack );
X              ftimer[i1] = time(&ttmp);
X              rack = (1 + rack) % MAXSEQ;
X              goto nloop;
X         }
X         if (dflg)
X              return(POK);
X         goto reply; /* any other stuff ? */
X
X    case EMPTY:
X         printmsg( 5, "**got EMPTY" );
X         itmp = time(&ttmp);
X         if (acktmr)
X              if ((itmp - acktmr) >= TIMEOUT) { /* ack timed out*/
X                   gspack(ACK, rwl, 0, 0, rdata);
X                   acktmr = itmp;
X              }
```

```
X        if (naktmr)
X             if ((itmp - naktmr) >= TIMEOUT) { /*nak timed out*/
X                  gspack(NAK, rwl, 0, 0, rdata);
X                  naktmr = itmp;
X             }
X        /* resend any timed out un-acked pkts */
X        for (i2 = swl; between(swl, i2, swu); i2 = (1 + i2) % MAXSEQ)  {
X             acktmr = naktmr = 0; /* reset ack/nak */
X             i1 = i2;
X             printmsg( 5, "--->seq,elapst %d %ld", i2, (itmp - ftimer[i1]) );
X             if ((itmp - ftimer[i1]) >= TIMEOUT) {
X                  printmsg( 5, "***timeout %d", i2 );
X                  /* since "g" is "go-back-N", when we time out we */
X                  /* must send the last N pkts in order. The generalized*/
X                  /* sliding window scheme relaxes this reqirment */
X                  nerr++;
X                  dflg = 1;  /* same hack */
X                  rack = i2;
X                  goto nloop;
X             }
X        }
X        return(POK);
X   case ACK:
X        printmsg( 5, "**got ACK %d", rack );
X        acktmr = naktmr = 0; /* disable ack/nak's */
Xaloop:
X        if (between(swl, rack, swu)) {   /* S<-- -->(S+W-1)%8 */
X             printmsg( 5, "***ACK %d", swl );
X             ftimer[swl] = 0;
X             nbuffers--;
X             swl = (1 + swl) % MAXSEQ;
X             dflg = 1;  /* same hack */           /* sl */
X             goto aloop;
X        }
X        if (dflg)
X             return(POK); /* hack for non-mtask sys's */
X        /* to empty "inbuf[]" */
X        goto reply;
X   case DATA:
X        printmsg( 5, "**got DATA %d %d", rack, rseq );
X        i1 = (1 + rwl) % MAXSEQ; /* (R+1)%8 <-- -->(R+W)%8 */
X        i2 = (1 + rwu) % MAXSEQ;
X        if (between(i1, rseq, i2)) {
X             if (i1 == rseq) {
X                  i1 = rseq;
X                  arr[i1] = TRUE;
X                  inlen[i1] = rlen;
X                  strncpy(inbuf[i1], rdata, rlen);
X                  rwl = (rwl + 1) % MAXSEQ;
X                  printmsg( 5, "***ACK d %d", rwl );
X                  gspack(ACK, rwl, 0, 0, rdata);
X                  acktmr = time(&ttmp); /* enable ack/nak tmout*/
X                  dflg = 1; /*return to call when finished*/
```

```
X                  /* in a mtask system, unneccesary */
X             } else {
X                  nerr++;
X                  printmsg( 5, "***unexpect %d ne %d", rseq, rwl );
X             }
X        } else {
X             nerr++;
X             printmsg( 5, "***wrong seq %d", rseq );
X        }
X        goto aloop;
X    case ERROR:
X        nerr++;
X        printmsg( 5, "**got BAD CHK" );
X        gspack(NAK, rwl, 0, 0, rdata);
X        naktmr = time(&ttmp); /* set nak timer */
X        printmsg( 5, "***NAK d %d", rwl );
X        goto reply;
X    default:
X        printmsg( 5, "**got SCREW UP" );
X        goto reply; /* ignore it */
X    }
Xclose:
X    gspack(CLOSE, 0, 0, 0, rdata);
X    return(CLOSE);
X}
X
X
X/**/
X/*************** FRAMMING ****************************/
X/*
X *
X *
X *      send a packet
X * nt2=type nt3=pkrec nt4=pksent len=length<=PKTSIZE cnt1= data * ret(0) always
X */
Xgspack(nt2, nt3, nt4, len, cnt1)
Xint nt2, nt3, nt4, len;
Xchar    cnt1[];
X{
X    unsigned int  check, i;
X    unsigned char c2, pkt[HDRSIZE+1], dpkerr[10];
X    if (len > 64)
X        len = 64;
X    if (len == 0)
X        cnt1[0] = '\0';
X/**Link testing mods- create artificial errors ***/ /*
Xprintf("**n:normal,e:error,l:lost,p:partial,h:bad header,s:new seq--> ");
Xgets(dpkerr);
X    if(dpkerr[0] == 's') { sscanf(&dpkerr[1],"%d",&nt4); } /**/
X    /** End Link testing mods ***/
X    printmsg( 5, "send packet type %d, num=%d, n=%d, len=%d", nt2, nt3, nt4,
len );
X    printmsg( 5, "data =\n|%s|", cnt1 );
```

```
X    c2 = '\0';
X    pkt[0] = '\020';
X    pkt[4] = nt2 << 3;
X    nt2 &= 7;
X    switch (nt2) {
X    case 1:
X         break;                  /* stop protocol */
X    case 2:
X         pkt[4] += nt3;
X         break;   /* reject        */
X    case 3:
X         break;
X    case 4:
X         pkt[4] += nt3;
X         break;   /* ack          */
X    case 5:
X         pkt[4] += SWINDOW;
X         break;   /* 3 windows */
X    case 6:
X         pkt[4] += 1;
X         break;   /* pktsiz = 64 (1) */
X    case 7:
X         pkt[4] += SWINDOW;
X         break;   /* 3 windows */
X    case 0:
X         pkt[4] += 0x80 + nt3 + (nt4 << 3);
X         c2 = (PKTSIZE - len) & 0xff;
X         /* havnt set it up for VERY LONG pkts with a few */
X         /* bytes yet (-128) */
X         if (c2) { /* short packet handling */
X              pkt[4] += 0x40;   /* if len < PKTSIZE */
X              for (i = PKTSIZE - 1; i > 0; i--)
X                   cnt1[i] = cnt1[i-1];
X              cnt1[0] = c2;
X         }
X         break;
X    }
X    pkt[4] &= 0xff;
X    if (nt2) {
X         pkt[1] = 9;             /* control packet size = 0 (9) */
X         check = (0xaaaa - pkt[4]) & 0xffff;
X    } else {
X         pkt[1] = PKTSIZ2;             /* data packet size = 64 (2) */
X         check = checksum(cnt1, PKTSIZE);
X         i = pkt[4];/* got to do this on PC for ex-or high bits */
X         i &= 0xff;
X         check = (check ^ i) & 0xffff;
X         check = (0xaaaa - check) & 0xffff;
X    }
X    pkt[2] = check & 0xff;
X    pkt[3] = (check >> 8) & 0xff;
X    pkt[5] = (pkt[1] ^ pkt[2] ^ pkt[3] ^ pkt[4]) & 0xff;
X/***More Link testing MODS ******/ /*
```

```
Xswitch(dpkerr[0]) {
Xcase 'e':   cnt1[10] = -cnt1[10];
X             break;
Xcase 'h':    pkt[5] = -pkt[5];
X             break;
Xcase 'l':    return;
Xcase 'p':    swrite(pkt,HDRSIZE);
X             if(pkt[1] != 9) swrite(cnt1,PKTSIZE-3);
X             return;
Xdefault: break;
X    }  /**/
X    /******End Link Testing Mods **********/
X    swrite(pkt, HDRSIZE);       /* header is 6-bytes long */
X    /*      write(flog,pkt,HDRSIZE);   */
X    if (pkt[1] != 9) {
X         swrite(cnt1, PKTSIZE);     /* data is always 64
X         bytes long */
X         /*    write(flog,cnt1,PKTSIZE);   */
X    }
X}
X
X
X/**/
X/*
X *
X *      read packet
X * on return: nt3=pkrec nt4=pksent len=length <=PKTSIZE  cnt1=data *
X * ret(type) ok; ret(EMPTY) input buf empty; ret(ERROR) bad header;
X *          ret(EMPTY) lost pkt timeout; ret(ERROR) checksum error;ret(-5) ?
X****NOTE :
X***sread(buf,n,timeout)
X    while(TRUE) {
X         if(# of chars available >= n) (without dec internal counter)
X              read n chars into buf (decrenent internal char counter)
X              break
X         else
X              if(time>timeout) break
X    }
X    return(# of chars available)
X****END NOTE
X */
Xgrpack(nt3, nt4, len, cnt1)
Xint *nt3, *nt4, *len;
Xchar    cnt1[];
X{
X    unsigned int  nt1, check, checkchk, i;
X    unsigned char c, c2;
X    int  ii;
X    if (GOT_SYNC)
X         goto get_hdr;
X    if (GOT_HDR)
X         goto get_data;
X    c = '\0';
```

```
X    while ((c & 0x7f) != '\020')
X         if (sread(&c, 1, timeout) == 0)
X              return(EMPTY);
X    GOT_SYNC = TRUE;
Xget_hdr:
X    if (sread(&grpkt[1], HDRSIZE - 1, timeout) < (HDRSIZE - 1))
X         return(EMPTY);
X    GOT_SYNC = FALSE;
X    /*i = grpkt[1] ^ grpkt[2] ^ grpkt[3] ^ grpkt[4] ^ grpkt[5];*/
X    i = (unsigned)grpkt[1] ^ (unsigned)grpkt[2] ^
X        (unsigned)grpkt[3] ^ (unsigned)grpkt[4] ^
X        (unsigned)grpkt[5];
X
X    i &= 0xff;
X    printmsg( 10, "prpkt %02x %02x %02x %02x %02x .. %02x ",
X                  grpkt[1], grpkt[2], grpkt[3], grpkt[4], grpkt[5],
X                  i
X             );
X
X    if (i) {               /*  bad header */
X         printmsg( 0, "****bad header****" );
X         return(ERROR); /* Im not sure whether "g" considers it an empty or error
*/
X    }
X    GOT_HDR = TRUE;
X    if ((grpkt[1] &= 0x7f) == 9) {       /* control packet */
X         *len = 0;
X         c = grpkt[4] & 0xff;
X         nt1  = c >> 3;
X         *nt3 = c & 7;
X         *nt4 = 0;
X         check = 0;
X         checkchk = 0;
X         cnt1[*len] = '\0';
X         GOT_HDR = FALSE;
X    } else {       /* data packet */
X         if (grpkt[1] != PKTSIZ2)
X              return(-5);   /* cant handle other than 64*/
Xget_data:
X         if (sread(cnt1, PKTSIZE, timeout) < PKTSIZE)
X              return(EMPTY);
X         GOT_HDR = FALSE;
X         nt1  = 0;
X         c2 = grpkt[4] & 0xff;
X         c = c2 & 0x3f;
X         *nt4 = c >> 3;
X         *nt3 = c & 7;
X         i = grpkt[3];
X         i = (i << 8) & 0xff00;
X         check = grpkt[2];
X         check = i | (check & 0xff);
X         checkchk = checksum(cnt1, PKTSIZE);
X         i = grpkt[4] | 0x80;
```

```
X        i &= 0xff;
X        checkchk = 0xaaaa - (checkchk ^ i);
X        checkchk &= 0xffff;
X        if (checkchk != check) {
X             printmsg( 4, "***checksum error***" );
X             return(ERROR);
X        }
X        *len = PKTSIZE;
X        /* havnt set it up for very long pkts yet (>128) RH Lamb */
X        if (c2 & 0x40) {
X             ii = (cnt1[0] & 0xff);
X             *len = (*len - ii) & 0xff;
X             for (ii = 0; ii < *len; ii++)
X                  cnt1[ii] = cnt1[ii+1];
X        }
X        cnt1[*len] = '\0';
X    }
X    printmsg( 5, "rec  packet type %d, num=%d, n=%d, len=%d", nt1, *nt3, *nt4,
*len );
X    printmsg( 6, "  checksum rec = %x comp = %x, data=\n|%s|", check, checkchk,
cnt1 );
X    ii = nt1;
X    return(ii);
X}
X
X
Xunsigned checksum(data, len)
Xint len;
Xchar    data[];
X{
X    unsigned int  i, j, tmp, chk1, chk2;
X    chk1 = 0xffff;
X    chk2 = 0;
X    j = len;
X    for (i = 0; i < len; i++) {
X        if (chk1 & 0x8000) {
X             chk1 <<= 1;
X             chk1++;
X        } else {
X             chk1 <<= 1;
X        }
X        tmp = chk1;
X        chk1 += (data[i] & 0xff);
X        chk2 += chk1 ^ j;
X        if ((chk1 & 0xffff) <= (tmp & 0xffff))
X             chk1 ^= chk2;
X        j--;
X    }
X    return(chk1 & 0xffff);
X}
X
X
X/* */
```

```
X/*
Xgwrmsg
X    send a null terminated string out
X*/
Xgwrmsg( typ, buf )
Xchar typ;
Xchar *buf; /* null terminated */
X{
X}
X
X/*
Xgrdmsg
X    read a null terminated string
X*/
Xgrdmsg( buf )
Xchar * buf;
X{
X
X}
X
X/*
Xgwrdata
X    read a file and send it out
X*/
Xgwrdata( f )
X{
X}
X
X/*
Xgrrdata
X    read in data and send to file
X*/
Xgrrdata( f )
X{
X}
X
X
X/*
Xgrdblk
X    read a block of data in
X*/
Xgrdblk( blk, len )
X{
X}
X
X/*
Xgwrblk
X    write out a block of data
X*/
Xgwrblk( blk, len )
X{
X}
X
```

```
SHAR_EOF
chmod 0600 dcpgpkt.c || echo "restore of dcpgpkt.c fails"
sed 's/^X//' << 'SHAR_EOF' > dcpsys.c &&
X
X/*           dcpsys.c
X
X             Revised edition of dcp
X
X             Stuart Lynne May/87
X
X             Copyright (c) Richard H. Lamb 1985, 1986, 1987
X             Changes Copyright (c) Stuart Lynne 1987
X
X*/
X/* "DCP" a uucp clone. Copyright Richard H. Lamb 1985,1986,1987 */
X/* Get the next system, and other support routines  */
X#include "dcp.h"
X/*#define PROTOS  "trkg"*/
X#define PROTOS  "g"
X#define MAXLOGTRY       3
X
XProto Protolst[] = {
X         'g', ggetpkt, gsendpkt, gopenpk, gclosepk,
X/*
X        'k', kgetpkt, ksendpkt, kopenpk, kclosepk,
X        'r', rgetpkt, rsendpkt, ropenpk, rclosepk,
X        't', tgetpkt, tsendpkt, topenpk, tclosepk,
X*/
X          '0'};
X
X#define EOTMSG "\004\r\004\r"
X
Xprocref         getpkt, sendpkt, openpk, closepk;
X
X/**/
X
X/*****************************************************************/
X/***            Sub Systems             */
X/*
X**
X**getsystem
X** Process an "systems" file entry (like L.sys)
X*/
Xgetsystem()
X{
X    int  i;
X
X    do {
X         if ( ! fgets( sysline, BUFSIZ, fsys ) )
X              return( 'A' );
X         printmsg( 2, "%s", sysline );
X    } while (sysline[0] == '#');
X
```

```
X    kflds = getargs( sysline, flds );
X    strcpy( rmtname, flds[FLD_REMOTE] );
X    cctime = flds[FLD_CCTIME];
X    strcpy( device, flds[FLD_DEVICE] );
X
X    /* strcpy( type, flds[FLD_TYPE] ); */
X    strcpy( speed, flds[FLD_SPEED] );
X    strcpy( proto, flds[FLD_PROTO] );
X
X
X    if (debuglevel > 3)
X         for (i = FLD_EXPECT; i < kflds; i += 2)
X              fprintf( stderr, "expect[%02d]:\t%s\nsend  [%02d]:\t%s\n",
X                   i, flds[i], i+1, flds[i+1] );
X
X    printmsg( 2, "rmt= %s ctm= %s", rmtname, flds[FLD_CCTIME] );
X    printmsg( 2, "dev= %s ", device );
X    printmsg( 2, "spd= %s pro= %s", speed, proto );
X    fw = (FILE *)NULL;
X    if (
X         /* (checktime( cctime )) || */
X         ( strncom( Rmtname, "all", 4 ) == SAME ) ||
X         ( strncom( Rmtname, rmtname, -1 ) == SAME ) ||
X         ( (strncom( Rmtname, "any", 4 ) == SAME) && scandir() == 'S' )
X         )
X    {
X         if ( fw != (FILE *)NULL )
X           fclose( fw );    /* in case we matched with scandir */
X         return( 'S' );  /* startup this system */
X    }
X    else
X         return('G');
X}
X
X/*
X**
X**checktime
X** check if we may make a call at this time
X**------------>to be implemented. Again. Didnt think it crucial
X*/
Xchecktime(xtime)
Xchar    xtime[];
X{
X    return(0); /* OK go to it */
X}
X
X
X
X/**/
X/*
X**
X**sysend
X** end UUCP session negotiation
```

```
X*/
Xsysend()
X{
X    char msg[80];
X
X    msg[1] = '\0';
X    msgtime = 2 * MSGTIME;
X    /* while (msg[1] != 'O') { */
X         wmsg("OOOOOO", 2);
X         if (rmsg(msg, 2) == -1)
X              goto hang;
X    /*}*/
Xhang:
X    wmsg("OOOOOO", 2);
X    closeline();
X    if ( remote == MASTER )
X         return('I');
X    return('A');
X}
X
X
X/**/
X/*
X**
X**      delay
X**
X*/
X/*ddelay(dtime)
Xint dtime;
X{
X    int  i, j;
X    for (i = 0; i < dtime; i++) {
X    }
X}
X*/
X
X/**/
X/*
X**
X**wmsg
X** write a ^P type msg to the remote uucp
X*/
Xwmsg(msg, syn)
Xchar    *msg;
Xint syn;
X{
X    int  len;
X    len = strlen(msg);
X    if (syn == 2)
X         swrite("\020", 1);
X    swrite(msg, len);
X    if (syn == 2)
X         swrite("\n", 2);
```

```
X}
X
X
X/*
X**
X**rmsg
X** read a ^P msg from UUCP
X*/
Xrmsg(msg, syn)
Xint syn;
Xchar    msg[];
X{
X    int  ii;
X    char c, cc[5];
X    /* *msg0;*/
X    /*msg0 = msg;*/
X    c = 'a';
X    if (syn == 2) {
X         while ((c & 0x7f) != '\020') {
X              if (sread(cc, 1, msgtime) < 1)
X                   return(-1);
X              c = cc[0]; /* Dont ask. MSC needs more than a byte to breathe */
X              /*       printf("Hello im in rmsg c=%x\n",c); */
X         }
X    }
X    for (ii = 0; ii < 132 && c ; ii++) {
X         if (sread(cc, 1, msgtime) < 1)
X              return(-1);
X         c = cc[0] & 0x7f;
X         if (c == '\r' || c == '\n')
X              c = '\0';
X         msg[ii] = c;
X         /*if(c == '\020') msg = msg0; */
X    }
X    return(strlen(msg));
X}
X
X
X
X/**/
X/*
X**
X**
X**startup
X**
X**
X*/
Xstartup()
X{
X    char msg[80], tmp1[20], tmp2[20];
X    if ( remote == MASTER ) {
X         msgtime = 2 * MSGTIME;
X         if (rmsg(msg, 2) == -1)
```

```
X             return('Y');
X         printmsg( 2, "1st msg = %s", msg );
X         if (msg[5] == '=' && strncmp(&msg[6], rmtname, 7))
X             return('Y');
X
X         /*sprintf(msg, "S%.7s -Q0 -x%d", nodename, debuglevel);*/  /* -Q0 -x16
remote debuglevel set */
X         sprintf(msg, "S%.7s", nodename);
X
X         wmsg(msg, 2);
X         if (rmsg(msg, 2) == -1)
X             return('Y');
X         printmsg( 2, "2nd msg = %s", msg );
X         if (strncmp(&msg[1], "OK", 2))
X             return('Y');
X         if (rmsg(msg, 2) == -1)
X             return('Y');
X         printmsg( 2, "3rd msg = %s", msg );
X         if (msg[0] != 'P' || index(&msg[1], proto[0]) == (char *)NULL) {
X             wmsg("UN", 2);
X             return('Y');
X         }
X         sprintf(msg, "U%c", proto[0]);
X         wmsg(msg, 2);
X         setproto(proto[0]);
X         return('D');
X    } else {
X         msgtime = 2 * MSGTIME;
X         sprintf(msg, "Shere=%s", nodename);
X         wmsg(msg, 2);
X         if (rmsg(msg, 2) == -1)
X             return('Y');
X         sscanf(&msg[1], "%s %s %s", rmtname, tmp1, tmp2);
X         sscanf(tmp2, "-x%d", &debuglevel);
X         printmsg( 1, "debuglevel level = %d", debuglevel );
X         printmsg( 2, "1st msg from remote = %s", msg );
X         if (checkname(rmtname))
X             return('Y');
X         wmsg("ROK", 2);
X         sprintf(msg, "P%s", PROTOS);
X         wmsg(msg, 2);
X         if (rmsg(msg, 2) == -1)
X             return('Y');
X         if (msg[0] != 'U' || index(PROTOS, msg[1]) == (char *)NULL )
X             return('Y');
X         proto[0] = msg[1];
X         setproto(proto[0]);
X         return('R');
X    }
X}
X
X
X/******* set the protocol **********/
```

```
Xsetproto(pr)
Xchar    pr;
X{
X    int  i;
X    Proto * tproto;
X    for (tproto = Protolst; tproto->type != '\0' && pr != tproto->type; tproto++)
{
X         printmsg( 3, "setproto: %c %c", pr, tproto->type );
X    }
X    if (tproto->type == '\0') {
X         printmsg( 0, "setproto:You said I had it but I cant find it" );
X         exit(1);
X    }
X    getpkt = tproto->a;
X    sendpkt = tproto->b;
X    openpk = tproto->c;
X    closepk = tproto->d;
X}
X
X
X
X/**/
Xint prefix(sh,lg)
Xchar *sh,*lg;
X{
X    return( strncmp(sh,lg,strlen(sh)) == SAME);
X}
X
Xint notin(sh,lg)
Xchar *sh,*lg;
X{
X    while (*lg) {
X         if (prefix(sh,lg++))
X              return( FALSE );
X    }
X    return( TRUE );
X}
X
X#define MAXR 300
Xint expectstr( str, timeout )
Xchar *str;
X{
X    static char rdvec[MAXR];
X    char *rp = rdvec;
X    int kr;
X    char nextch;
X
X    printmsg( 0, "wanted %s", str );
X
X    if ( strcmp(str, "\"\"") == SAME ) {
X         return( TRUE );
X    }
X    *rp = 0;
```

```
X    while ( notin( str,rdvec ) ) {
X         /* fprintf(stderr, "---------->%s<------\n", rdvec);/**/
X         kr = sread( &nextch, 1, timeout /* 40 */ );
X         /* nextch &= 0177;
X         fprintf(stderr, "kr - %2d '%c'\n", kr, nextch); */
X         if (kr <= 0) {
X              return( FALSE );
X         }
X         if ((*rp = nextch & 0177) != '\0') {
X              rp++;
X         }
X         *rp = '\0';
X         if (rp >= rdvec + MAXR) {
X              return( FALSE );
X         }
X    }
X    return( TRUE );
X}
X
Xint writestr(s)
X    register char *s;
X{
X    register char last;
X    register char * m;
X    int nocr;
X    last = '\0';
X    nocr = FALSE;
X    while (*s) {
X         if (last == '\\') {
X              switch (*s) {
X              case 'd':
X              case 'D': /* delay */
X                sleep(2);
X                break;
X             case 'c':
X             case 'C': /* end string don't output CR */
X                nocr = TRUE;
X                break;
X             case 'r':
X             case 'R': /* carriage return */
X             case 'm':
X             case 'M':
X                  swrite( "\r", 1 );
X                  break;
X             case 'n':
X             case 'N':
X                  swrite( "\n", 1 );
X                  break;
X             case 'b':
X             case 'B':
X                  swrite( "\b", 1 );
X                  break;
X             case 't':
```

```
X            case 'T':
X                 swrite( "\t", 1 );
X                 break;
X            case 's':
X            case 'S':
X                 swrite( " ", 1 );
X                 break;
X            case 'z':
X            case 'Z':
X             SIOSpeed( ++s );
X             while ( *s != '\0' && *s != '\\' )
X                  s++;
X             if ( *s == '\\' )
X                  s++;
X             break;
X            default:
X                 swrite( s, 1 );
X             }
X             last = '\0';
X        }
X        else if (*s != '\\') {
X             swrite( s, 1 );
X             /* fputc(*s,stderr); */
X        }
X        else {
X             last = *s;
X        }
X        s++;
X    }
X    return( nocr );
X}
X
X/***
X *   void sendthem(str)   send line of login sequence
X *         char *str;
X *
X *   return codes:  none
X */
X
Xvoid sendstr(str)
Xchar *str;
X{
X    int nw, ns;
X    int nulls;
X
X    printmsg( 2, "sending %s", str );
X
X#ifdef BREAK
X    if (prefix("BREAK", str)) {
X         sscanf(&str[5], "%1d", &nulls);
X         if (nulls <= 0 || nulls > 10)
X              nulls = 3;
X         /* send break */
```

```
X         ssendbrk(nulls);
X         return;
X    }
X#endif
X
X    if ( strcmp(str, "EOT") == SAME ) {
X         swrite(EOTMSG, strlen(EOTMSG));
X         return;
X    }
X
X    if ( strcmp(str,"\"\"") == SAME )
X         *str = '\0';
X         /*fprintf(stderr,"'%s'\n",str);*/
X
X    if ( strcmp(str,"") != SAME ) {
X         if (!writestr(str)) {
X              swrite ("\r", 1);
X         }
X    }
X    else {
X         swrite("\r", 1);
X    }
X    return;
X}
X
Xint sendexpect( s, e, timeout )
Xchar * s;
Xchar * e;
X{
X    sendstr( s );
X    return( expectstr( e, timeout ) );
X}
X
Xdial()
X{
X    int  flg, kk, jj, ll, firstflg;
X    char buf[4], *prsend;
X
X    char *exp;
X    char *alternate;
X    int  ok;
X    int  doofosity=0;
X    int i;
X
X    if ( strcmp( flds[FLD_TYPE], "HAYES" ) != SAME ) {
X         printmsg( 0, "dial: unsupported dialer %s", flds[FLD_TYPE] );
X         return( FALSE );
X    }
X
X    printmsg( 3, "calling host %s", rmtname );
X    if (openline(device, "2400" ))
X         return( FALSE );
X/*
```

```
X    printmsg( 0, "hayes: trying 2400" );
X    if ( sendexpect( "\\r\\dATZ", "OK", 4 ) != TRUE ) {
X         sendexpect( "\\d+++\\d", "OK", 2 );
X         if ( sendexpect( "ATZ", "OK", 2 ) != TRUE ) {
X              printmsg( 0, "hayes: trying 1200" );
X              SIOSpeed( "1200" );
X              if ( sendexpect( "ATZ", "OK", 2 ) != TRUE ) {
X                   sendexpect( "\\d+++\\d", "OK", 2 );
X                   if ( sendexpect( "ATZ", "OK", 2 ) != TRUE )
X                        return( FALSE);
X                   }
X         }
X    }
X    printmsg( 0, "hayes: got modem response" );
X*/
X    /*(sendstr( "\\d\\dATS7=30" );
X    expectstr( "OK", 40 );*/
X
X    sendstr( "\\dATX4\\c" );
X    sendstr( speed );
X
X    if ( sendexpect( "", "CONNECT ", 40 ) == TRUE ) {
X         printmsg( 3, "hayes: got CONNECT" );
X
X         if ( sread( buf, 4, 4 ) == 4 ) {
X              printmsg( 0, "hayes: speed select %s", buf );
X              /* set speed appropriately */
X              SIOSpeed( buf );
X         }
X         return( TRUE );
X    }
X    return( FALSE );
X}
X
X
X/*
X**
X**callup
X** script processor - nothing fancy!
X*/
Xcallup()
X{
X    int  flg, kk, jj, ll, firstflg;
X    char *prsend;
X
X    char *exp;
X    char *alternate;
X    int  ok;
X    int i;
X
X    printmsg( 0, "calling host %s", rmtname );
X
X    if ( strcmp( flds[FLD_TYPE], "DIR" ) != SAME ) {
```

```
X         if ( dial() == FALSE )
X              return( 'G' );
X    }
X    else if (openline(device, speed))
X         return( 'G' );
X
X    for (i = 6; i < kflds; i+=2) {
X
X         exp = flds[i];
X         printmsg( 2, "callup: expect %d of %d  \"%s\"", i, kflds, exp );
X
X         ok = FALSE;
X         while (ok != TRUE) {
X              int time = 30;
X              char *timel;
X
X              timel = index( exp, '~' );
X              if (timel != (char *)NULL)
X              {
X                   time = atoi(timel+1);
X                   alternate = timel+1;
X                   while (isdigit(*alternate))
X                        alternate++;
X                   strcpy(timel, alternate);
X              }
X
X              alternate = index( exp, '-' );
X              if (alternate != (char *)NULL)
X                   *alternate++ = '\0';
X
X              ok = expectstr( exp, time );
X
X              printmsg( 1, "got %s", ok != TRUE ? "?" : "that" );
X
X              if ( ok == TRUE ) {
X                   printmsg( 0, "got that" );
X                   break;
X              }
X
X              if ( alternate == (char *)NULL ) {
X                   printmsg( 0, "LOGIN FAILED" );
X                   return( 'Y' );
X              }
X
X              exp = index( alternate, '-' );
X              if ( exp != (char *)NULL )
X                   *exp++ = '\0';
X
X              printmsg( 0, "send alternate" );
X
X              sendstr( alternate );
X         }
X
```

```
X    printmsg( 2, "callup: send %d of %d  \"%s\"", i+1, kflds, flds[i+1] );
X    sleep(1); /* (1)*/
X    sendstr(flds[i+1]);
X    }
X    return('P');
X
X}
X
X/**/
X/*
X**
X**      slowrite
X** comunication slow write. needed for auto-baud modems
X*/
X/*slowrite(st)
Xregister char    *st;
X{
X    int  len, j;
X    char c;
X    len = strlen(st);
X    printmsg( 2, "sent %s", st );
X    for (j = 0; j < len; j++) {
X         swrite(&st[j], 1);
X         ddelay(80000);
X    }
X}
X*/
X
X/**/
X/*
X**
X**scandir
X**
X*/
X
X#include "ndir.h"
X
X
X/*  scandir
X
X    scan work dir for C. files matching current remote host (rmtname)
X
X    return
X
X         A    - abort
X         Y    - can't open file
X         S    - ok
X         Q    - no files
X
X*/
Xscandir()
X{
X    int  fn, len, i;
```

```
X    char cname[40], tmp[132];
X
X    DIR *dirp;
X    struct direct *dp;
X
X    if ((dirp = opendir( spooldir )) == (DIR *)NULL ) {
X         fprintf( stderr, "couldn't open dir %s\n", spooldir );
X         return( 'A' );
X    }
X    sprintf(cname, CALLFILE, rmtname);
X    len = strlen(cname);
X    while ((dp = readdir(dirp)) != (struct direct *)NULL) {
X         printmsg( 4, "scandir: %s", dp->d_name );
X         if ( strncom( cname, dp->d_name, len ) == SAME ) {
X              printmsg( 4, "scandir: match!!" );
X              strcpy(cfile, dp->d_name);
X              closedir( dirp );
X              if ((fw = FOPEN( cfile, "r", 't' )) == (char *)NULL )
X                   return('Y');
X              return('S');
X         }
X    }
X    closedir( dirp );
X    return('Q');
X
X}
X
X
X/**/
X/*
X**
X**dscandir
X** scan the directory
X*/
X
Xdscandir()
X{
X    int  fn, len, i;
X    char cname[40], tmp[132];
X
X    DIR *dirp;
X    struct direct *dp;
X
X
X    if ((dirp = opendir( spooldir )) == (DIR *)NULL ) {
X         fprintf( stderr, "couldn't open dir %s\n", spooldir );
X         return(0);
SHAR_EOF
echo "End of part 1, continue with part 2"
echo "2" > s2_seq_.tmp
exit 0
```

From 1058973397820403658
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,71f6207310ae1f2a,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-07-29 10:07:56 PST
Path: utzoo!utgpu!water!watmath!clyde!att!osu-cis!tut.cis.ohio-state.edu!husc6!
bbn!uwmcsd1!ig!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i066: uupc -- UUCP clone (pre-pre-release) part02/04
Keywords: Laser, buggy
Message-ID: <4333@saturn.ucsc.edu>
Date: 29 Jul 88 17:07:56 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 2047
Approved: koreth@ssyx.ucsc.edu

Submitted-by: koreth@ssyx.ucsc.edu (Steven Grimm)
Posting-number: Volume 1, Issue 66
Archive-name: uupc/part02

```
#!/bin/sh
# this is part 2 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file dcpsys.c continued
#
CurArch=2
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> dcpsys.c
X    }
X    sprintf(cname, XQTFILE, rmtname); /* sprintf(cname,"c%.4s",rmtname); */
X    len = strlen(cname);
X    while ((dp = readdir(dirp)) != (struct direct *)NULL) {
X         printmsg( 4, "dscandir: file = %s cname = %s", dp->d_name, cname );
X         if ( strncmp( cname, dp->d_name, len ) == SAME ) {
X              printmsg( 4, "dscandir: match!!" );
X              strcpy(cfile, dp->d_name);
X              closedir( dirp );
X              return( -1 );
X         }
X    }
X    closedir( dirp );
X    return( 0 );
X
X}
```

```
X
X
X
X
SHAR_EOF
chmod 0600 dcpsys.c || echo "restore of dcpsys.c fails"
sed 's/^X//' << 'SHAR_EOF' > dcpxfer.c &&
X/*           dcpxfer.c
X
X             Revised edition of dcp
X
X             Stuart Lynne May/87
X
X             Copyright (c) Richard H. Lamb 1985, 1986, 1987
X             Changes Copyright (c) Stuart Lynne 1987
X
X*/
X/* "DCP" a uucp clone. Copyright Richard H. Lamb 1985,1986,1987 */
X/* file send routines */
X#include "dcp.h"
X#include <ctype.h>
X
Xstatic unsigned char  rpacket[MAXPACK];
Xstatic unsigned char  spacket[MAXPACK];
X/**/
X/***************SEND PROTOCOL*****************************/
X/*
X *  s d a t a
X *
X *  Send File Data
X */
Xsdata()
X{
X
X    while( TRUE ) {
X
X         if (( *sendpkt) ( spacket, size, 0 ))
X              return (0 );     /* send data */
X         if (( size = bufill( spacket )) == 0 )  /* Get data from file */
X              return( 'Z' );                          /* If EOF set state to
that */
X    }
X
X    return('D');/* Got data, stay in state D */
X
X}
X
X
X/*
X *  b u f i l l
X *
X *  Get a bufferful of data from the file that's being sent.
X *  Only control-quoting is done; 8-bit & repeat count prefixes are
```

```
X *  not handled.
X */
Xbufill(buffer)
Xchar    *buffer;/* Buffer */
X{
X    return( read(fp, buffer, pktsize) );/* Handle partial buffer */
X}
X
X
X/*
X *  s b r e a k
X *
X *  Send Break (EOT)
X */
Xsbreak()
X{
X    int  len, i;
X    sprintf(spacket, "H");
X    if ((*sendpkt)(spacket, 0, 1))
X         return(0);
X    if ((*getpkt)(spacket, &len))
X         return(0);
X    printmsg( 2, "Switch modes" );
X    if (spacket[1] == 'N')
X         return('G');
X    return('Y');
X}
X
X
X/**/
X/*
X *  s e o f
X *
X *  Send End-Of-File.
X */
Xseof()
X{
X    int  len, i;
X    if ((*sendpkt)(spacket, 0, 0))
X         return(0);
X    if ((*getpkt)(spacket, &len))
X         return(0); /* rec CY or CN */
X    if (strncmp(spacket, "CY", 2))
X         return(0); /* cant send file */
X    close(fp);
X    fp = (-1);
X    importpath( hostfile, fromfile );
X    unlink(hostfile);
X    printmsg( 0, "Transfer of %s (%s) completed.  fp=%d", fromfile, hostfile,
fp );
X
X    /*
X    fprintf( syslog, "%s!%s (%d/%d-%d:%d:%d) -> %ld / %ld secs", host, id, size,
```

```
secs );
X    */
X
X    return('F');                    /* go get the next file to send */
X}
X
X
X/**/
X/*
X *  s f i l e
X *
X *  Send File Header.
X */
Xsfile()
X{
X    int  i, len;
X    char * cp;
X    if (fp == -1) {/* If not already open, */
X         printmsg( 3, "looking for next file..." );
X         if (getfile()) { /* get next file from current work*/
X              fclose( fw );
X              unlink( cfile );/* close and delete completed workfile */
X              fw = (char *)NULL;
X              return('B'); /* end sending session */
X         }
X
X         importpath( hostfile, fromfile );
X
X         printmsg( 3, "Opening %s (%s) for sending.", fromfile, hostfile );
X         fp = open(hostfile, 0);/* open the file to be sent */
X         if (fp == -1) {/* If bad file pointer, give up */
X              printmsg( 0, "Cannot open file %s (%s).", fromfile, hostfile );
X              return('A');
X         }
X    } else
X         return('A'); /* If somethings already open. were in trouble*/
X    printmsg( 0, "Sending %s (%s) as %s", fromfile, hostfile, tofile );
X    strcpy(spacket, tofile);
X    if ((*sendpkt)(spacket, 0, 1))
X         return(0);       /* send S fromfile tofile */
X    if ((*getpkt)(spacket, &len))
X         return(0);       /* user - tofile 0666. */
X    if (spacket[1] != 'Y')
X         return('A');           /* If otherside says no-quit*/
X    size = bufill(spacket);
X    return('D');
X}
X
X
X/**/
X/*
X *  s i n i t
X *
```

```
X *  Send Initiate: send this host's parameters and get other side's back.
X */
Xsinit()
X{
X    if ((*openpk)())
X         return('A');
X    return('B');
X}
X
X
X/**/
X/*
X *
X *
X *  getfile
X *
X *  getfile reads the next line from the presently open workfile
X *  (cfile) and determines from this the next file to be sent
X *  (file). If there are no more TRUE is returned.
X *  --A fix for "R from to 0666" should be done here to recieve files
X *  --in addition to sending them. The appropriate "state letter"
X *  --i.e. "R" should be returned to the send "master" or "slave"
X *  --state switching table in "dcp.c"
X *  --I did not implement this since the majority of uucp transactions
X *  --appear to be "S from to 0666" type. RHLamb 1/87
X *
X */
Xgetfile()
X{
X    int  i;
X    char line[132];
X    register char * cp;
X
X    if ( ! fgets( line, BUFSIZ, fw ) )
X         return(TRUE);
X
X    sscanf(&line[2], "%s ", fromfile);
X    for ( i = 0, cp = line; *cp!='\0'; i++, cp++ ) {
X         if ( strncmp( cp, "0666", 4 ) == 0)
X              break;
X    }
X    cp+=4;
X    *cp = '\0';
X    strcpy(tofile, line);
X    printmsg(3, "   getfile: fromfile=%s, tofile=%s.", fromfile, tofile);
X    return(FALSE);
X}
X
X
X/**/
X/********************** MISC SUB SUB PROTOCOL *************************/
X/*
X**
```

```
X**schkdir
X** scan the dir
X*/
Xschkdir()
X{
X    char c;
X    c = scandir();
X    if (c == 'Q') {
X         return('Y');
X    }
X    if (c == 'S') {
X         sprintf(rpacket, "HN");
X         if ((*sendpkt)(rpacket, 0, 1))
X              return(0);
X    }
X    return('B');
X}
X
X
X/**/
X/*
X *
X *      endp() end protocol
X *
X*/
Xendp()
X{
X    sprintf(rpacket, "HY");
X    (*sendpkt)(rpacket, 0, 2); /* dont wait for ACK */
X    (*closepk)();
X    return('P');
X}
X
X
X/**/
X/***********************RECIEVE PROTOCOL**********************/
X/*
X *  r d a t a
X *
X *  Receive Data
X */
Xrdata()
X{
X    int  len;
X    if ((*getpkt)(rpacket, &len))
X         return(0);
X    if (len == 0) {
X         close(fp);
X         sprintf(rpacket, "CY");
X         if ((*sendpkt)(rpacket, 0, 1))
X              return(0);
X         printmsg( 0, "transfer complete" );
X         return('F');
```

```
X    }
X    write(fp, rpacket, len);/* Write the data to the file */
X    return('D');/* Remain in data state */
X}
X
X
X/**/
X/*
X *  r f i l e
X *
X *  Receive File Header
X */
Xrfile()
X{
X
X    char     buf[256];
X    char *   flds[10];
X    int      numflds;
X
X    int      len, i;
X    char tmpfilename[256]; /*Holds the converted file name */
X    char *cp;
X
X    printmsg( 3, "rfile entered" );
X
X    cp = buf;
X
X    while ( TRUE ) {
X         if ((*getpkt)( rpacket, &len ))
X              return( 0 );
X         strncpy( cp, rpacket, len );
X         cp += len;
X         if ( *(cp - 1) == '\0' ) break;
X    }
X    if (( buf[0] & 0x7f ) == 'H' )
X         return( 'C' );
X
X    printmsg( 3, "rfile: buf %d \"%s\"", len, buf );
X
X    /* Convert upper case to lower */
X    for (cp = buf; *cp != '\0';cp++)
X         if (isupper(*cp)) tolower(*cp);
X
X    numflds = getargs( buf, flds );
X
X    cp = flds[2];
X    printmsg( 3, "rfile: receive file \"%s\"", cp );
X
X    /* check for ~/ destination -> /usr/spool/uucppublic */
X    if ( strncmp( cp, "~/", 2 ) == SAME )
X         sprintf( tmpfilename, "%s%s", pubdir, cp+1);
X    else
X         strcpy( tmpfilename, cp );
```

```
X    printmsg( 3, "rfile: receive file \"%s\"", tmpfilename );
X
X    /* check for dirname only */
X    cp = tmpfilename + strlen( tmpfilename ) - 1;
X    if ( *cp == '\n' )
X         *cp-- = '\0';
X
X    if ( *cp == '/' ) {
X
X         fprintf( stderr, "rfile: fromfile %s\n", flds[1] );
X         cp = rindex( flds[1], '/' );
X         if ( cp == (char *) NULL )
X              cp = flds[1];
X         else
X              cp++;
X
X         fprintf( stderr, "rfile: dironly add   %s\n", cp );
X
X         strcat( tmpfilename, cp );
X    }
X    printmsg( 3, "rfile: receive file \"%s\"", tmpfilename );
X
X
X    /* let host munge filename as appropriate */
X    importpath( tofile, tmpfilename );
X    printmsg( 3, "rfile: receive file \"%s\"", tofile );
X
X    if ((fp = CREAT( tofile, 0775, 'b' )) == -1) { /* Try to open a new file */
X         printmsg( 0, "cannot create %s", tofile ); /* Give up if can't */
X         return('A');
X    }
X
X    printmsg( 0, "Receiving %s as %s", flds[1], tofile );
X    sprintf(rpacket, "SY");
X    if ((*sendpkt)(rpacket, 0, 1))
X         return(0);
X    return('D'); /* Switch to data state */
X}
X
X
X/**/
X/*
X *  r i n i t
X *
X *  Receive Initialization
X */
Xrinit()
X{
X    if ((*openpk)())
X         return(0);
X    return('F');
X}
X
```

```
X
X
X
SHAR_EOF
chmod 0600 dcpxfer.c || echo "restore of dcpxfer.c fails"
sed 's/^X//' << 'SHAR_EOF' > dirsubs.c &&
X/*
X * getcwd(), chdir()
X */
X#include <osbind.h>
X#define NULL 0
X
Xextern char *malloc();
Xextern char errno;
X
Xchar *getcwd( pnbuf, maxlen )
Xchar *pnbuf;
Xint  maxlen;
X{
X    int drive;
X
X    if ( pnbuf == (char *)NULL )
X    {
X       if ( (pnbuf = malloc( 64 )) == (char *)NULL )
X       {
X          printf( "getcwd: malloc failure\n" );
X          exit(-1);
X       }
X    }
X
X    drive = Dgetdrv(); /* get the current drive */
X    pnbuf[0] = (char)('A'+drive);
X    pnbuf[1] = ':';
X
X    Dgetpath( &pnbuf[2], 0 ); /* get current directory for current drive */
X
X    return( pnbuf );
X}
X
Xint chdir( path )
Xchar *path;
X{
X    int status;
X    /*
X     * Change current drive if the path has a drive specified
X     */
X    if ( path[1] == ':' )
X       Dsetdrv( (int)(path[0]-'A') );
X
X    status = Dsetpath( path );  /* set current path */
X    if ( status == 0 ) return(0);
X    printf("chdir: failed with error %d on %s\n", status, path);
X    errno = status;
```

```
X    return(-1);
X}
X
X
SHAR_EOF
chmod 0600 dirsubs.c || echo "restore of dirsubs.c fails"
sed 's/^X//' << 'SHAR_EOF' > dmymath.c &&
X/*
X * ftoa, etoa, atof - Dummy routines so linker does not complain.
X * Will report error and exit should they actually be called.
X */
X
Xdouble atof( nptr )
Xchar *nptr;
X{
X    printf(" atof: Not implemented\n" );
X    exit(-1);
X}
X
Xchar *etoa( fval, buf, prec )
Xdouble fval;
Xchar *buf;
Xint prec;
X{
X    printf(" etoa: Not implemented\n" );
X    exit(-1);
X}
X
Xchar *ftoa( fval, buf, prec )
Xdouble fval;
Xchar *buf;
Xint prec;
X{
X    printf(" ftoa: Not implemented\n" );
X    exit(-1);
X}
X
X
SHAR_EOF
chmod 0600 dmymath.c || echo "restore of dmymath.c fails"
sed 's/^X//' << 'SHAR_EOF' > fgets.c &&
X#include <stdio.h>
X#include <osbind.h>
X#include <ctype.h>
X
X#define TRUE 1
X#define FALSE 0
X
X#define CR '\015'
X#define LF '\012'
X#define BS '\010'
X#define CD '\004'
X#define CU '\025'
```

```
X#define CX '\030'
X#define CZ '\032'
X
Xstatic char backsp[] = "\010 \010";
Xstatic char newline[]= "\015\012";
X
X/*
X * Fgets() - A replacement for fgets() that does its own console I/O
X *      since the library version seems not to work.
X *      If the I/O is not to the console then we just call the
X *      real fgets().
X *
X *  To use you must define fgets as Fgets either at the top of your
X *  source file or in a common header file.
X */
X
Xchar *Fgets( buf, n, stream )
Xchar *buf;
Xint n;
XFILE *stream;
X{
X    static int eof = FALSE;
X    int fd,eol;
X    char *s = buf;
X    char c;
X
X    eol = FALSE;
X    n  -= 1; /* Leave room for the null character */
X
X    fd = fileno( stream );
X    if ( isatty( fd ) )
X    {
X       /*
X        * If the user typed ^Z last time then report EOF on this call.
X        * Clear EOF as well so that future calls to the console will
X        * work.
X        */
X       if ( eof )
X       {
X          eof = FALSE;
X          return( (char *)NULL );
X       }
X       /*
X        * Not EOF so read up to n character and return the line.
X        */
X       while ( ( s - buf ) < n )
X       {
X           c = ( Crawcin() & 0xFF );
X           /*
X            * If its a printing character then just echo it
X            */
X           if (isprint( c ))
X           {
```

```
X             Cconout( c );
X             *s++ = c;
X             continue;
X          }
X          /*
X           * It must be a control character, either process it or
X           * ignore it.
X           */
X          switch (c)
X          {
X             case CD:
X             case CZ:
X                eof = eol = TRUE;
X                if ( s == buf ) break;
X             case CR:
X             case LF:
X                eol = TRUE;
X                *s++ = LF;
X                if ( ( s - buf ) < n ) *s++ = '\0';
X                Cconws( newline );
X                break;
X             case BS:
X                if ( s != buf )
X                {
X                   s--;
X                   Cconws( backsp );
X                }
X                break;
X             case CU:
X             case CX:
X                while ( s != buf )
X                {
X                   s--;
X                   Cconws( backsp );
X                }
X             default:
X                break;
X          }
X          if ( eol ) break;
X       }
X       if ( eof && s == buf )
X       {
X          eof = FALSE;
X          return( (char *)NULL );
X       }
X       return( buf );
X    }
X    /*
X     * Was not console so call regular fgets() since it seems to
X     * work for everything else.
X     */
X    return( fgets( buf, n, stream ) );
X}
```

```
X
X#ifdef DEB_Fgets
X
Xextern int errno;
Xchar buf[200];
X
X#define fgets Fgets /* Use our own fgets routine */
X
Xmain ( argc, argv )
Xint argc;
Xchar **argv;
X{
X
X    char *s;
X
X    while ( ( s = fgets( buf, 20, stdin ) ) != (char *)NULL )
X    {
X       puts(">>");
X       puts( s );
X       puts("<<");
X    }
X    return( errno );
X}
X#endif
X
X
SHAR_EOF
chmod 0600 fgets.c || echo "restore of fgets.c fails"
sed 's/^X//' << 'SHAR_EOF' > genv.c &&
X/*  genv.c
X
X
X
X    copyright (C) 1987 Stuart Lynne
X
X    Copying and use of this program are controlled by the terms of the
X    Free Software Foundations GNU Emacs General Public License.
X
X
X    version       0.1      March 31/1987
X
X
X
X*/
X
X#include <stdio.h>
X
X#ifndef NULL
X#define NULL 0
X#endif
X
X#include "genv.h"
X/*
```

```
X#define DMAILBOX  "mailbox"
X#define  DNAME          "noname"
X#define  DDOMAIN        "pc.uucp"
X#define  DMAILDIR  "\\usr\\spool\\mail"
X#define DHOME          "\\usr\\noname"
X#define  DCONFDIR  "\\usr\\lib\\uucp"
X#define  DSPOOLDIR"\\usr\\spool\\uucp"
X#define DLOGDIR        "\\usr\\spool\\uucp"
X#define DPUBDIR        "\\usr\\spool\\uucppublic"
X#define DNEWSSDIR "\\usr\\spool\\rnews"
X#define DTEMPDIR  "\\tmp"
X#define  DMAILSERVICE  "host"
X#define  DNODENAME"noname"
X#define  DDEVICE        "AUX:"
X#define  DSPEED         "1200"
X*/
X
X#define  TFILENAME"tmpfile"
X#define  FILENAME  "%s\\%s"
X
Xchar *getenv();
Xchar *malloc();
Xvoid genv();
X
Xextern int debuglevel;
X
X#ifdef NOMALLOC
Xchar    rmailbox[64];
Xchar    rname[64];
Xchar    rdomain[64];
Xchar    rmaildir[64];
Xchar    rhome[64];
Xchar    rconfdir[64];
Xchar    rlogdir[64];
Xchar    rspooldir[64];
Xchar    rpubdir[64];
Xchar    rnewssdir[64];
Xchar    rtempdir[64];
Xchar    rmailserv[16];
Xchar    rnodename[16];
Xchar    rdevice[16];
Xchar    rspeed[64];
X
Xchar    *mailbox = rmailbox;
Xchar    *name = rname;
Xchar    *domain = rdomain;
Xchar    *maildir = rmaildir;
Xchar    *home = rhome;
Xchar    *confdir = rconfdir;
Xchar    *spooldir = rspooldir;
Xchar    *pubdir = rpubdir;
Xchar    *logdir = rlogdir;
Xchar    *newssdir = rnewssdir;
```

```
Xchar    *tempdir = rtempdir;
Xchar    *mailserv = rmailserv;
Xchar    *nodename = rnodename;
Xchar    *device = rdevice;
Xchar    *speed = rspeed;
X#else
Xchar    *mailbox;
Xchar    *name;
Xchar    *domain;
Xchar    *maildir;
Xchar    *home;
Xchar    *confdir;
Xchar    *spooldir;
Xchar    *logdir;
Xchar    *pubdir;
Xchar    *newssdir;
Xchar    *tempdir;
Xchar    *mailserv;
Xchar    *nodename;
Xchar    *device;
Xchar    *speed;
X#endif
X
Xvoid genv(thename, envname, dflt)
Xchar **thename;
Xchar *envname;
Xchar *dflt;
X{
X    char *temp;
X
X    if ((temp = getenv( envname )) == (char *)NULL) {
X       if ( debuglevel > 0 )
X          fprintf( stderr, "genv: %s not found, using %s\n",
X                           envname, dflt );
X       temp = dflt;
X    }
X    *thename = malloc( strlen( temp ) + 1 );
X    strcpy( *thename, temp );
X    if ( debuglevel > 0 )
X       fprintf( stderr, "genv: %s = %s\n", envname, *thename );
X}
X
Xvoid loadenv()
X{
X    /* get environment var's */
X    genv( &name, "NAME", DNAME );
X    genv( &mailbox, "MAILBOX", DMAILBOX );
X    genv( &domain, "DOMAIN", DDOMAIN );
X    genv( &maildir, "MAILDIR", DMAILDIR );
X    genv( &confdir, "CONFDIR", DCONFDIR );
X    genv( &spooldir, "SPOOLDIR", DSPOOLDIR );
X    genv( &logdir, "LOGDIR", DLOGDIR );
X    genv( &home, "HOME", DHOME );
```

```
X    genv( &pubdir, "PUBDIR", DPUBDIR );
X    genv( &newssdir, "NEWSSDIR", DNEWSSDIR );
X    genv( &tempdir, "TEMPDIR", DTEMPDIR );
X    genv( &mailserv, "MAILSERVICE", DMAILSERVICE );
X    genv( &nodename, "NODENAME", DNODENAME );
X    genv( &device, "DEVICE", DDEVICE );
X    genv( &speed, "SPEED", DSPEED );
X}
X
Xvoid mkfilename( filename, dirname, name )
Xchar *filename;
Xchar *dirname;
Xchar *name;
X{
X    sprintf( filename, FILENAME, dirname, name );
X    if ( debuglevel > 5 )
X       fprintf( stderr, "genv: New filename %s\n", filename );
X}
X
X
X#ifdef TEST
Xmain()
X{
X    loadenv();
X}
X#endif
X
X
SHAR_EOF
chmod 0600 genv.c || echo "restore of genv.c fails"
sed 's/^X//' << 'SHAR_EOF' > genv.h &&
X/*  genv.c
X
X
X
X    copyright (C) 1987 Stuart Lynne
X
X    Copying and use of this program are controlled by the terms of the
X    Free Software Foundations GNU Emacs General Public License.
X
X
X    version       0.1      March 31/1987
X
X
X
Xenvironment variables
X
X    The following evironment variables are defined:
X
X    MAILBOX       current user's mailbox,     "sl"
X    NAME     current user's name,        "Stuart Lynne"
X    DOMAIN        domain of this machine,     "mac.van-bc.can"
X    MAILDIR       where mail is kept,         "/usr/mail"
```

```
X    CONFDIR       where config data is  kept, "/usr/lib/uucp"
X    SPOOLDIR  where spooled files are     "/usr/spool/uucp"
X    LOGDIR        where log file are placed   "/usr/spool/uucp"
X    PUBDIR        public spool directory      "/usr/spool/uucppublic"
X    NEWSSDIR  news spool directory        "/usr/spool/rnews"
X    TEMPDIR       where temp files are created     "/tmp"
X    HOME     where uses private files are     "/usr/sl"
X    MAILSERVICE   who do we send remote mail  "van-bc"
X    NODENAME  what is our local nodename  "slmac"
X    DEVICE        what is the default device  ".a"
X    SPEED         what is the default speed   "1200"
X
X*/
X
X
X#define  MAILBOX       "koreth"
X#define  NAME      "Steven Grimm"
X#define  DOMAIN        "holodek.uucp"
X#define  MAILDIR       "D:\\USR\\SPOOL\\MAIL"
X#define HOME      "D:\\USR\\KORETH"
X
X#define  CONFDIR       "D:\\UUPC"
X#define SPOOLDIR  "D:\\USR\\SPOOL\\UUPC"
X#define LOGDIR        "D:\\USR\\SPOOL\\UUPC"
X#define  PUBDIR        "D:\\USR\\SPOOL\\UUPC"
X#define  NEWSSDIR  "D:\\USR\\SPOOL\\NEWS"
X#define TEMPDIR       "D:\\TMP"
X#define  MAILSERVICE   "ssyx"
X
X#define  DMAILBOX  "koreth"
X#define  DNAME         "Steven Grimm"
X#define  DDOMAIN       "holodek.uucp"
X#define  DMAILDIR  "D:\\USR\\SPOOL\\MAIL"
X#define DHOME         "D:\\USR\\KORETH"
X
X#define  DCONFDIR  "D:\\UUPC"
X#define DSPOOLDIR "D:\\USR\\SPOOL\\UUPC"
X#define DLOGDIR       "D:\\USR\\SPOOL\\UUPC"
X#define  DPUBDIR       "D:\\USR\\SPOOL\\UUPC"
X#define  DNEWSSDIR"D:\\USR\\SPOOL\\NEWS"
X#define DTEMPDIR  "D:\\TMP"
X#define  DMAILSERVICE  "ssyx"
X
X#define NODENAME  "holodek"
X#define  DEVICE        "AUX:"
X#define  SPEED         "2400"
X
X#define DNODENAME "holodek"
X#define  DDEVICE       "AUX:"
X#define  DSPEED        "2400"
X
Xextern char  *mailbox;
Xextern char  *name;
```

```
Xextern char  *domain;
Xextern char  *maildir;
Xextern char  *home;
X
Xextern char  *confdir;
Xextern char  *spooldir;
Xextern char  *newssdir;
Xextern char  *pubdir;
Xextern char  *tempdir;
Xextern char  *mailserv;
X
Xextern char  *nodename;
Xextern char  *device;
Xextern char  *speed;
X
Xvoid loadenv();
Xvoid genv();
Xvoid mkfilename();
X
X
SHAR_EOF
chmod 0600 genv.h || echo "restore of genv.h fails"
sed 's/^X//' << 'SHAR_EOF' > getargs.c &&
X#include <ctype.h>
X
Xint getargs( line, flds )
Xchar *line;
Xchar **flds;
X{
X    int i = 0;
X    char *s;
X
X    while ( (*line != '\0') && (*line != '\n') )
X    {
X       if ( isspace(*line) )
X       {
X          line++;
X          continue;
X       }
X       *flds++ = line;
X       i++;
X       while( (isspace(*line) == 0) && (*line != '\0') ) line++;
X       if (isspace(*line)) *line++ = '\0';
X    }
X    return(i);
X}
X
X#ifdef TEST
X
Xchar comline[64] = "this is a test\n";
X
Xmain(argc, argv)
Xint argc;
```

```
Xchar *argv;
X{
X    char *flds[60];
X    int  i,j;
X
X    i = getargs( comline, flds );
X
X    printf( "Num args = %d\n", i );
X
X    for (j=0; j<i; j++)
X      printf( "Arg[%d] = %d %s\n", j, strlen(flds[j]), flds[j] );
X}
X#endif
X
X
X
SHAR_EOF
chmod 0600 getargs.c || echo "restore of getargs.c fails"
sed 's/^X//' << 'SHAR_EOF' > getenv.c &&
X/*
X * char *getenv( s )
X *
X */
X#include <string.h>
X#define strchr index
X#define strrchr rindex
X
Xextern char *_base;        /* pointer to base page */
X
X#ifndef NULL
X#define NULL 0
X#endif
X
Xchar *getenv( s )
Xchar *s;
X{
X   char *envp;
X   int  n;
X
X   envp = *(char **)(_base + 44);    /* pointer to enviroment strings */
X
X/*printf( "Base address = %lx, Enviroment = %lx\n", _base, envp ); /* */
X
X   if ( *envp == '\0' ) return (char *)NULL;
X
X   while ( *envp != '\0' )
X   {
X
X/*printf( "Checking %s\n", envp ); /* */
X
X      n = strlen(s);
X
X      if ( strncmp( envp, s, n ) != 0 )
```

```
X      {
X         while ( *envp++ != '\0' );
X/*printf( "Failed match, next string at %lx\n", envp ); /* */
X         continue;
X      }
X/*printf( "Matched %s\n", envp ); /* */
X      envp += n;              /* skip over variable name */
X      if ( *envp != '=' )
X      {
X         envp = strchr( envp, '\0' ) + 1; /* skip this string */
X         continue;
X      }
X      break;
X   }
X   return ( ( *envp != '=' ) ? (char *)NULL : ++envp );
X}
X
X#ifdef TEST
X
Xmain( argc, argv )
Xint argc;
Xchar **argv;
X{
X    char *s;
X
X    argc--;            /* skip program name */
X    argv++;
X
X    while ( argc-- )
X    {
X      s = *argv;
X      while ( *s != '\0' )
X      {
X        if ( *s >= 'a' && *s <= 'z' ) *s = *s - 32;
X        *s++;
X      }
X      s = getenv( *argv );
X      if (s == (char *)NULL)
X        printf( "%s is not defined\n", *argv );
X      else
X        printf( "%s = %s\n", *argv, s );
X      argv++;
X    }
X}
X#endif
X
X
SHAR_EOF
chmod 0600 getenv.c || echo "restore of getenv.c fails"
sed 's/^X//' << 'SHAR_EOF' > getone.c &&
X#include <osbind.h>
X
Xint get_one()
```

```
X{
X    int c;
X
X    c = Crawcin() & 0xFF;
X    if ( c == '\r' ) c = '\n';
X    return( c );
X}
X
X
SHAR_EOF
chmod 0600 getone.c || echo "restore of getone.c fails"
sed 's/^X//' << 'SHAR_EOF' > host.c &&
X#include <stdio.h>
X#include "genv.h"
X
Xint  chdir();
Xvoid loadenv();
Xvoid printmsg();
X
Xextern int errno;
X
Xchar *curdir;
Xchar *getcwd();
Xchar c_curdir[256];
X
Xjmp_buf dcpexit;
X
X
Xmain( argc, argv )
Xint argc;
Xchar **argv;
X{
X    /* host specific prolog */
X    loadenv();
X    curdir = getcwd( c_curdir, 0 );
X    if (chdir( spooldir ) != 0)
X    {
X       return(errno);
X    }
X
X    /* setup longjmp for error exit's */
X    if ( setjmp( dcpexit ) == 0 ) {
X
X         MAIN(argc, argv );
X
X    }
X
X
X    /* host specific epilog */
X    if (chdir( curdir ) != 0)
X    {
X       return(errno);
X    }
```

```
X    return(0);
X}
X
X
SHAR_EOF
chmod 0600 host.c || echo "restore of host.c fails"
sed 's/^X//' << 'SHAR_EOF' > host.h &&
X/* Atari ST, Alcyon C Version 4.14a                    */
X/* Host specific defines for uu and mail               */
X
X#include <ctype.h>
X#include <strings.h>
X#include <osbind.h>
X#include "time.h"
X#include "genv.h"
X
X#define XENIX         /* don't send debugging info in startup message
     */
X#define fgets Fgets   /* use our own fgets to fix problem in runtime
     */
X#define mkdir(x) Dcreate(x)     /* create directory
              */
X
X#define ATARIST
X
X#define strchr  index
X#define strrchr rindex
X#define FILEMODE(f)
X
X#define SEPCHAR       '\\' /* path separater character
     */
X#define XQTFILE   "X_%.6s" /* prefix for execute files
     */
X#define CALLFILE  "C_%.6s" /* prefix for call files
     */
X#define TFILENAME "TM.%03ld"    /* template for temporary files
     */
X#define SFILENAME "SEQF"        /* name of the sequence file
         */
X#define SIGFILE       "SIGNATUR"    /* file containing signature
     */
X#define COPYFILE  "MAIL\\MAILSENT"   /* copy of sent mail
     */
X#define NEWSDIR       "\\usr\\spool\\rnews\\%02d%02d%02d%02d.%03d"
     /* format for news name*/
X#define HOSTINIT      /* name of function used to initialize
     */
X
X/* Rsconf() defines */
X
X#define RSR       (-1)      /* leave as is              */
X#define TSR       (-1)      /* " "      " "             */
X#define SCR       (-1)      /* " "      " "             */
```

```
X#define UCR       (0x88)        /* 8-bits, no parity, 2 stops
    */
X#define NOFLOW        (0)       /* no flow control     */
X
X#define B19200        (0)       /* 19200 baud          */
X#define B9600         (1)       /* 9600 baud           */
X#define B4800         (2)       /* 4800 baud           */
X#define B2400         (4)       /* 2400 baud           */
X#define B1200         (7)       /* 1200 baud           */
X#define B300      (9)       /*  300 baud           */
X
X/* misc. defines */
X
X#ifndef TRUE
X#define FALSE         (0)
X#define TRUE      (-1)
X#endif
X
X#ifndef SAME
X#define SAME      (0)
X#endif
X
X#define AUX       (1)       /* rs232 port          */
X#define CON       (2)       /* console             */
X
X#define CTRL(X)       (X & 037) /* get a control character
    */
X
X#define  IBUFSIZ      (0x1000)
X#define  OBUFSIZ      (0x1000)
X
X/* typedef for Iorec() stuff (modified & improved by Martin Minow) */
X
Xstruct iorec
X  {
X    char*ibuf;        /* Pointer to input buffer */
X    int      ibufsize;/* Size of input buffer    */
X    int      ibufhd;      /* Head index              */
X    int      ibuftl;      /* Tail index         */
X    int      ibuflow; /* Low water mark     */
X    int      ibufhi;      /* Hi water mark      */
X  };
X
Xtypedef struct {
X    struct iorec in;          /*  0 Input buffer     */
X    struct iorec out;         /* 14 Output buffer
    */
X} IOREC;
X
X
X/*
X * Rsconf() returns the "old" parameters in a longword encoded as follows:
X * bits      value
```

```
X * 31-24 UCR
X * 23-16 RSR
X * 15- 8 TSR
X *  8- 0 SCR
X *
X * Note that we must redefine the Rsconf() macro.
X */
X
X
X#define  RS_UCR_SHIFT  24
X#define RS_RSR_SHIFT  16
X#define RS_TSR_SHIFT   8
X#define RS_SCR_SHIFT   0
X
Xextern int errno;
X
X
SHAR_EOF
chmod 0600 host.h || echo "restore of host.h fails"
sed 's/^X//' << 'SHAR_EOF' > inoutpth.c &&
X#include <strings.h>
X#include <ctype.h>
X
X#define NULL 0
X#define strchr index
X#define strrchr rindex
X
X#define min(x,y) (((x) < (y)) ? (x) : (y))
X/*
X *                  Canonical name conversion routines
X *
X * These routines convert file name between canonical form which is defined
X * as a 'unix' style pathname and whatever format the host system can
X * handle.
X */
X
X/*
X * importpath( host, canon )
X *
X * Convert a canonical name to a format the host can handle
X *
X * Mung the canonical file name as follows:
X *   1 - strip any path from the canonical name
X *   2 - copy up to 8 character from the canonical name converting . to _
X *   3 - if the name was longer than 8 character copy a . to the host name
X *       and then copy the up to three characters from the tail of the
X *   canonical name to the host name.
X */
Xchar *importpath( host, canon )
Xchar *host, *canon;
X{
X    char *s,*out,c;
X    int  i,j,l;
```

```
X
X    out  = host;
X
X    /*
X     * get a pointer to the last component of the path
X     */
X
X    if ( (s = strrchr( canon, '/' )) == (char *)NULL ) s = canon;
X    else s++;
X
X    j = min(l = strlen(s),8);
X
X    for (i=0; i<j; i++)
X    {
X       c = *s++;
X       *out++ = (c == '.') ? '_' : c;
X    }
X    *out = '\0';
X
X    while (*s != '\0') s++;
X
X    if (l>8)
X       for (i=0; i<3; i++)
X          if (*--s == '.') { s++; break; }
X
X    if (*s != '\0')
X    {
X
X       strcat( out, "." );
X       strcat( out, s );
X    }
X
X    return( host );
X}
X
X/*
X * exportpath( canon, host )
X *
X * Convert a host name into canonical name
X *
X * Copy file name from host to canon converting an underscore in the second
X * character to a dot and any other dots into the letter B.
X *
X */
Xexportpath( canon, host )
Xchar *host, *canon;
X{
X    register char c;
X    register char *s = canon;
X    do
X    {
X       c = *s = tolower( *host );
X       if ( c == '.' ) *s = 'B';
```

```
X       s++;
X    } while ( *host++ != '\0' );
X
X    *canon = toupper( *canon );
X    if ( canon[1] == '_' )
X       canon[1] = '.';
X}
X
X
SHAR_EOF
chmod 0600 inoutpth.c || echo "restore of inoutpth.c fails"
sed 's/^X//' << 'SHAR_EOF' > lib.c &&
X/*      lib.c
X
X*/
X
X#include <stdio.h>
X#include <ctype.h>
X#include "host.h"
X
X#ifndef NULL
X#define NULL 0L
X#endif
X
Xchar *index();
Xchar *rindex();
X
XMKDIR( path )
Xchar * path;
X{
X    char * cp = path;
X
X    if ( *cp == '\0' )
X        return( 0 );
X
X    /* see if we need to make any intermediate directories */
X    while ( ( cp = index( cp, '\\' ) ) != (char *) NULL ) {
X        *cp = '\0';
X        mkdir( path );
X        *cp = '\\';
X        cp++;
X    }
X
X    /* make last dir */
X    return( mkdir( path ) );
X
X}
X
XCHDIR( path )
Xchar * path;
X{
X    char * cp = path;
X
```

```
X    if ( *cp == '\0' )
X         return( 0 );
X
X    MKDIR( path );
X
X    /* change to last directory */
X    return( chdir( path ) );
X
X}
X
XFILE * FOPEN( name, mode, ftype )
Xchar * name;
Xchar * mode;
Xchar ftype;
X{
X
X    char * last;
X    FILE * results;
X
X    /* are we opening for write or append */
X
X    FILEMODE( ftype );
X    results = fopen( name, mode );
X
X    if ( results != (FILE *) NULL || *mode == 'r' )
X         return( results );
X
X    /* are we opening in sub-directory */
X    last = rindex( name, '/' );
X
X    /* lets just verify that all sub-dir's exist */
X    if ( last != (char *) NULL ) {
X         *last = '\0';
X         MKDIR( name );
X         *last = '/';
X    }
X
X    /* now try open again */
X    return( fopen( name, mode ));
X
X}
X
Xint CREAT( name, mode, ftyp )
Xchar * name;
Xint mode;
Xchar ftyp;
X{
X
X    char * last;
X    int results;
X
X    /* are we opening for write or append */
X    FILEMODE( ftyp );
```

```
X    results = creat( name, mode );
X
X    if ( results != -1 )
X        return( results );
X
X    /* are we opening in sub-directory */
X    last = rindex( name, '/' );
X
X    /* lets just verify that all sub-dir's exist */
X    if ( last != (char *) NULL ) {
X        *last = '\0';
X        MKDIR( name );
X        *last = '/';
X    }
X
X    /* now try open again */
X    return( creat( name, mode ) );
X
X}
X
Xextern FILE *logfile;
Xextern int debuglevel;
Xextern int remote;
X
X#define MASTER 1
X
X
Xint getargs( line, flds )
Xchar *line;
Xchar **flds;
X{
X    int i = 0;
X    char *s;
X
X    while ( (*line != '\0') && (*line != '\n') )
X    {
X       if ( isspace(*line) )
X       {
X          line++;
X          continue;
X       }
X       *flds++ = line;
X       i++;
X       while( (isspace(*line) == 0) && (*line != '\0') ) line++;
X       if (isspace(*line)) *line++ = '\0';
X    }
X    return(i);
X}
X
X
X
SHAR_EOF
chmod 0600 lib.c || echo "restore of lib.c fails"
```

```
sed 's/^X//' << 'SHAR_EOF' > lmail.c &&
X/*           lmail.c
X
X*/
X
X#define NOMAIN
X/* #define RMAIL /* if defined we get rmail() else we get lmail() */
X
X#include "pcmail.c"
SHAR_EOF
chmod 0600 lmail.c || echo "restore of lmail.c fails"
sed 's/^X//' << 'SHAR_EOF' > mail.c &&
X/*
X *  mail.c
X */
X
X#define  exit return
X
X#include <stdio.h>
X#include "host.h"
X
XFILE *FOPEN();
XFILE *freopen();
X
Xchar *index(), *rindex();
X
Xint debuglevel = 0;
X
X#define LSIZE    256
X
Xchar    line[LSIZE];
Xchar    resp[LSIZE];
X
Xchar    mfilename[132];
X
Xstruct ldesc {
X   int       delete;       /* status of this message */
X   long      adr;     /* address of From line */
X   long      date;         /* address of Date: line */
X   long      subject;  /* address of Subject: line */
X   long      from;         /* address of From: line */
X   long      size;         /* number of lines */
X};
X
Xstruct  ldesc letters[300];
X
Xint     letternum   = 0;
Xchar    thefilename[50];
X
Xchar    tmailbox[64];
Xchar    tmailbag[64];
X
X
```

```
XFILE    *fmailbox;
XFILE    *rmailbox;
XFILE    *fmailbag;
X
Xint error = 0;
Xint modified;
X
Xint printonly;
X
Xint PageCount = 0;
X
Xmailmain(argc, argv)
Xchar **argv;
X{
X
X    /* HOSTINIT;
X    loadenv(); */
X
X    mkfilename( tmailbox, tempdir, "mailbox");
X    mkfilename( tmailbag, tempdir, "mailbag");
X
X    unlink( tmailbox );
X
X    if ( mailbox == (char *)NULL )
X         mailbox = "???";
X
X    if ( argc == 1 ||
X         ( argv[1][0] == '-' ) && ( argv[1][1] != 's' ))
X         showmail (argc, argv );
X    else
X         lmail (argc, argv );
X
X    finis();
X
X}
X
Xfinis()
X{
X    if ( fmailbag != (FILE *)NULL) fclose( fmailbag );
X    unlink( tmailbox );
X    unlink( tmailbag );
X    exit( error );
X}
X
X/*  sendmail
X    send message n to "line"
X*/
Xsendmail( n, line )
Xchar *line;
X{
X    char     *argv[50];
X    int  argc;
X    char buf[BUFSIZ];
```

```
X    long     bytes;
X
X    /* open a temporary file */
X    fmailbag = FOPEN( tmailbag, "w", 'b' );
X
X    /* copy nth message to mailbag file */
X    fseek( fmailbox, letters[ n ].adr, 0 );
X    bytes = letters[ n+1 ].adr - letters[ n ].adr;
X
X    while( bytes > 0 && fgets( buf, BUFSIZ, fmailbox) ) {
X         fputs( buf, fmailbag  );
X         bytes -= strlen( buf );
X    }
X
X    fclose( fmailbag );
X
X    /* use mailbag as stdin to mail delivery */
X    fclose( stdin );
X    FILEMODE( 'b' );
X    if ( freopen( tmailbag, "r", stdin ) != (FILE *)NULL ) {
X         argc = getargs( line, argv );
X
X         lmail( argc, argv );
X
X         fclose( stdin );
X    }
X
X#ifdef MSDOS
X    FILEMODE( 't' );
X    freopen( CONSOLE, "r", stdin );
X#else
X    freopen( device, "r", stdin );
X#endif
X
X    /* get rid of mailbag */
X    unlink( tmailbag );
X}
X
X
Xinvert( i )
X{
X    return( letternum - i - 1 );
X}
X
X/*
X * Generate the return address for a letter by looking at the "From "
X * line.
X */
Xchar *returnaddress(let)
Xint let;
X{
X    struct ldesc *lp;
X    static char from[256];
```

```
X    char line[256], *i, *k;
X
X    lp = &letters[let];
X
X    readaline( lp->adr, line );
X
X    i = index(line+5, ' ');
X    if (i == NULL)
X         i = &line[strlen(line)];
X    strncpy( from, line+4, i-line-4 );
X    from[i-line-4] = '\0';
X
X    return from;
X}
X
X/*
X * Turn a subject into a reply (if it isn't already).  If there was no
X * subject line, return NULL.
X */
Xchar *re(let)
Xint let;
X{
X    static char newsub[256];
X    char subline[256];
X
X    subline[0] = '\0';
X
X    readaline(letters[let].subject, subline);
X    if (! subline[0])
X         return NULL;
X
X    if (subline[9] == 'R' && subline[10] == 'e' && subline[11] == ':')
X         strcpy(newsub, subline+9);
X    else
X    {
X         strcpy(newsub, "Re:  ");
X         strcat(newsub, subline+9);
X    }
X
X    return newsub;
X}
X
Xshowmail(argc, argv)
Xchar **argv;
X{
X    int flg, i, j, k, print;
X    char *p, *cp, *getnext();
X    long nextadr;
X    struct ldesc *letter;
X
X    /* get mailbox file name */
X    mkfilename( mfilename, maildir, mailbox );
X
```

```
X    /* parse arg's */
X    for (; argc > 1; argv++, argc--) {
X       if ( argv[1][0] == '-' ) {
X          if ( argv[1][1] == 'f' ) {
X             if ( argc >= 3) {
X                if ( argv[2][0] == '=' )
X                   mkfilename( mfilename, maildir, &argv[2][1] );
X                else
X                   strcpy( mfilename, argv[2] );
X             argv++;
X             argc--;
X             }
X
X          } else if ( argv[1][1 ]== 'p' ) printonly++;
X       else {
X          fprintf(stderr, "mail: unknown option %c\n", argv[1][1]);
X          finis();
X       }
X       } else break;
X    }
X
X
X
X    /* open real mailbox file */
X    if (( rmailbox = FOPEN( mfilename, "r", 'b' ))  == (FILE *)NULL) {
X         fprintf( stdout, "No mail in %s.\n", mfilename );
X         return;
X    }
X
X    /* open up tmp mailbox file */
X    if (( fmailbox = FOPEN( tmailbox, "w", 'b' )) == (FILE *)NULL ) {
X         fprintf( stderr, "mail: cannot open %s for writing\n", tmailbox );
X         finis();
X    }
X
X
X    /* copy real mailbox file to tmp mailbox file */
X    letternum = 0 ;
X    nextadr = 0;
X    while ( fgets(line, LSIZE, rmailbox ) ) {
X         fputs( line, fmailbox );
X
X         if ( strncmp( line, "From ", 5 ) == SAME ) {
X              letter = &letters[letternum++];
X              letter->from =
X              letter->subject =
X              letter->date = -1L;
X              letter->adr = nextadr;
X              letter->delete = FALSE;
X              letter->size = 0L;
X         }else if ( strncmp( line, "Date: ", 6 ) == SAME )
X              letter->date = nextadr;
X         else if ( strncmp( line, "From: ", 6 ) == SAME )
```

```
X            letter->from = nextadr;
X        else if ( strncmp( line, "Subject: ", 9 ) == SAME )
X            letter->subject = nextadr;
X
X        letter->size += 1L;
X        nextadr += strlen( line )+1;     /* +1 for \r\n */
X    }
X
X    letters[letternum].adr = nextadr;   /* last plus 1 */
X
X    /* close mailbox files */
X    fclose( rmailbox );
X    fclose( fmailbox );
X
X    fmailbox = FOPEN( tmailbox, "r", 'b' );
X
X    modified = 0;
X
X    if ( printonly ) {
X        j = letternum;
X        while ( j > 0 ) {
X            pager( j );
X            j--;
X        }
X
X        return;
X     }
X
X
X    printsub( -1 );
X
X    i = letternum - 1;
X    while ( TRUE ) {
X
X        if ( i < 0 )
X            i = 0;
X        if ( i >= letternum)
X            i = letternum - 1;
X        j = invert( i );
X
X        fprintf( stdout, "%d ? ", invert( i ) );
X        fflush(stdout);
X
X        if ( ! fgets( resp, LSIZE, stdin) )
X            break;
X
X        switch ( resp[0] ) {
X        default:
X            fprintf(stderr, "usage\n");
X        case '?':
X            print = 0;
X            fprintf( stderr, "q\tquit\n" );
X            fprintf( stderr, "x\texit\tmailbox restored\n" );
```

```
X             fprintf( stderr, "p\tprint message\n" );
X             fprintf( stderr, "s[file]\tsave message to a file (default =mbox)
\n" );
X             fprintf( stderr, "w[file]\ttsave message to a file (default =mbox)
without header\n" );
X             fprintf( stderr, "-\tprint last message\n" );
X             fprintf( stderr, "d\tdelete current message\n" );
X             fprintf( stderr, "+\tnext message (no delete)\n" );
X             fprintf( stderr, "f user\tforward current message to user\n" );
X             fprintf( stderr, "m user\tmail a message to user\n" );
X             fprintf( stderr, "r\treply to current message\n" );
X             fprintf( stderr, "g num\t set current message\n" );
X             break;
X         case '+':
X         case 'n':
X             i-- ;
X             break;
X         case 'g':
X             i = invert( atoi( resp+2 ) );
X             break;
X         case '0':
X         case '1':
X         case '2':
X         case '3':
X         case '4':
X         case '5':
X         case '6':
X         case '7':
X         case '8':
X         case '9':
X             i = invert( atoi( resp ) );
X             break;
X
X         case 'x':
X             modified = 0;
X         case 'q':
X             goto donep;
X         case '\n':
X         case 'p':
X             pager( j );
X             break;
X         case '^':
X         case '-':
X             i++;
X             break;
X         case 'y':
X         case 'w':
X         case 's':
X             flg = 0;
X             if (resp[1] != '\n' && resp[1] != ' ') {
X                 printf("illegal\n");
X                 flg++;
X                 print = 0;
```

```
X                continue;
X            }
X            if (strlen( resp ) == 1)
X                 strcat( resp, " =mbox" );
X            for (p = resp+1; (p = getnext( &cp, p )) != (char *)NULL; ) {
X               if ( *cp == '=' )
X                         mkfilename( thefilename, maildir, ++cp );
X               else
X                  strcpy( thefilename, cp );
X               fprintf( stderr, "%s to %s\n",
X                     (resp[0] == 's') ? "Appending":"Writing", thefilename );
X               if (( fmailbag=FOPEN( thefilename, "a", 't' )) == (FILE *)NULL )
X{
X                  fprintf( stdout, "mail: cannot append to %s\n",
X                           thefilename );
X               flg++;
X               continue;
X               }
X               copymsg( j, fmailbag, resp[0]=='w' );
X               fclose( fmailbag );
X            }
X            if (flg)
X                 print = 0;
X            else {
X                 letters[j].delete = TRUE;
X                 modified++;
X                 i--;
X            }
X            break;
X        case 'm':
X            mailto(resp+1, NULL);
X            break;
X        case 'r':
X            fprintf( stdout, "To: %s\n", returnaddress(j) );
X            mailto(returnaddress(j), re(j));
X            break;
X        case 'f':
X            flg = 0;
X            if (resp[1] == '\n' || resp[1] == '\0') {
X                 i++;
X                 continue;
X            }
X            if (resp[1] != ' ') {
X                 printf( "invalid command\n" );
X                 flg++;
X                 print = 0;
X                 continue;
X            }
X
X            sendmail( j, p );
X
X            if (flg)
X                 print = 0;
```

```
X             else {
X                  letters[j].delete = TRUE;
X                  modified++;
X                  i++;
X             }
X             break;
X
X        case 'd':
X             letters[j].delete = TRUE;
X             modified++;
X             i--;
X             if (resp[1] == 'q')
X                  goto donep;
X             break;
X
X        case 'h':
X             printsub( -1 );
X             break;
X
X        case 'H':
X             printsub( i );
X             break;
X
X        }
X
X    }
X    donep:
X    if ( modified )
X         copyback();
X}
X
X/*
X * Invoke lmail with a particular destination and, possibly, subject.
X */
Xmailto(who, what)
Xchar *who, *what;
X{
X    char sub[256];
X    char *argv[5];
X    int  argc = 1;
X
X    argv[0] = "lmail";
X    printf("Subject: ");
X    fflush(stdout);
X
X    if (what != NULL)
X         puts(what);
X    else
X         gets(what = sub);
X
X    putchar('\n');
X
X    argv[argc++] = "-s";
```

```
X    argv[argc++] = what;
X
X    while (*who == ' ')
X         who++;
X
X    argv[argc++] = who;
X
X    lmail(argc, argv);
X}
X
Xreadaline( adr, line )
Xlong adr;
Xchar *line;
X{
X    char     buf[132];
X    char *cp;
X
X    *line = '\0';
X    if ( adr != -1L )
X         if ( fseek( fmailbox, adr, 0 ) == 0 )
X              if ( fgets( buf, 132, fmailbox ) )
X                   strcpy( line, buf );
X
X    if ( (cp = index( line, '\n' ) ) != (char *) NULL)
X         *cp = '\0';
X}
X
X
Xprintsub( K )
X{
X    struct ldesc  *ld;
X    int      j;
X    char from[132], subject[132], date[132], line[132];
X    char *sp, *dp, *lp, *tp;
X    int k, mink, maxk;
X
X    pagereset();
X
X    if ( K == -1 ) {
X         maxk = letternum -1;
X         mink = 0;
X    }
X    else
X         maxk = mink = K;
X
X    for ( k = maxk ; k >= mink ; k-- ) {
X
X         j = invert( k );
X         ld = & letters[ j /*k*/];
X
X         (void)strcpy(from, "unknown");   /* default to "unknown" */
X         readaline( ld->from, line );
X         if ( strlen( line ) > 0 ) {
```

```
X            /* Assume From: mailbox@site.domain */
X            /* get first usable stuff after From: */
X            sp = index( line, ':' ) + 1;
X
X            /* do we have From: mailbox@site.domain ( Proper Name ) */
X            if ( ( ( tp = index( line, '(' )) != (char *) NULL &&
X                      ( lp = index( line, ')')) != (char *) NULL  ) )
X                 sp = ++tp;
X            /* or we may have From: Proper Name < mailbox@site.domain > */
X            else
X                 lp = index( line, '<');
X
X            if (lp != NULL)
X                 if ( *lp == '<' || *lp == ')' )
X                      --lp;
X
X            /* if we didn't get either of the last two, just grab line */
X            if ( lp < sp || lp == (char *) NULL )
X                 lp = line + strlen( line ) - 1;
X
X            /*debugMsg("\PCheck isspace *sp");*/
X            while ( isspace( *sp ) ) sp++;
X
X            /*debugMsg("\PCheck isspace *lp");*/
X            while ( isspace( *lp ) ) lp--;
X
X            dp = from;
X            while ( sp <= lp ) *dp++ = *sp++;
X            *dp = '\0';
X        }
X
X        /* Date: Wed May 13 23:59:53 1987 */
X        *date = '\0'; /* default date to null */
X        readaline( ld->date, date );
X        if ( strlen( date ) > 0 ) {
X            sscanf( date, "%*s %*s %s %s", line, subject );
X            sprintf( date, "%s %s", line, subject );
X        }
X
X        strcpy( subject, "--- no subject ---" );
X        readaline( ld->subject, line );
X        if ( strlen( line ) > 0 ) {
X            sp = line;
X            while ( !isspace( *sp ) ) sp++;
X            while ( isspace( *sp ) ) sp++;
X            strcpy( subject, sp );
X            }
X
X        from[20] = '\0';
X        date[6] = '\0';
X        subject[35] = '\0';
X
X        sprintf( line, "%3d  %6s  %20s %35s (%6d)\n",
```

```
X                           j, date, from, subject, ld->size );
X         if ( pageline( line ) == TRUE )
X              break;
X    }
X}
X
X
Xcopyback()
X{
X    register i, n, c;
X    int new = 0;
X
X    if (( fmailbag = FOPEN( mfilename, "w", 'b' )) == (FILE *)NULL ) {
X         fprintf (stderr, "mail: can't rewrite %s\n", mfilename );
SHAR_EOF
echo "End of part 2, continue with part 3"
echo "3" > s2_seq_.tmp
exit 0


From -2882214376817855
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,7b9f9398eaec2eca,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-07-29 10:08:59 PST
Path: utzoo!utgpu!water!watmath!clyde!att!osu-cis!tut.cis.ohio-state.edu!cwjcc!
gatech!rutgers!mailrus!uwmcsd1!ig!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i067: uupc -- UUCP clone (pre-pre-release)  part03/04
Keywords: Laser, buggy
Message-ID: <4334@saturn.ucsc.edu>
Date: 29 Jul 88 17:08:59 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1489
Approved: koreth@ssyx.ucsc.edu

Submitted-by: koreth@ssyx.ucsc.edu (Steven Grimm)
Posting-number: Volume 1, Issue 67
Archive-name: uupc/part03

#!/bin/sh
# this is part 3 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file mail.c continued
#
CurArch=3
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
```

```
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> mail.c
X        finis();
X    }
X    n = 0;
X    for ( i = 0; i < letternum; i++ )
X        if ( letters[i].delete == FALSE ) {
X             copymsg( i, fmailbag, FALSE );
X             n++;
X        }
X    fclose (fmailbag );
X}
X
X
X/* copy a message
X   if noheaders true, don't copy header lines
X*/
Xcopymsg( n, f, noheaders )
XFILE *f;
X{
X    long     bytes;
X    char buf[BUFSIZ];
X
X    fseek( fmailbox, letters[n].adr, 0 );
X    bytes = letters[n+1].adr - letters[n].adr;
X
X    while ( bytes > 0 && fgets( buf, BUFSIZ, fmailbox ) ) {
X        bytes -= strlen( buf );
X
X        /* write line */
X        if ( !noheaders  )
X             fputs( buf, f );
X
X        /* reset header */
X        if ( noheaders  && strcmp( buf, "\n") == SAME )
X             noheaders  = FALSE;
X
X    }
X}
X
Xpager(n)
X{
X    long     bytes;
X    char buf[BUFSIZ];
X
X    fseek( fmailbox, letters[n].adr, 0 );
X    bytes = letters[n+1].adr - letters[n].adr;
X
X    pagereset();
X    while ( bytes > 0 && fgets( buf, BUFSIZ, fmailbox ) ) {
X        bytes -= strlen( buf )+1;
```

```
X        if ( pageline( buf ) == TRUE )
X             break;
X    }
X
X    pageline( "\n" );
X}
X
Xpagereset()
X{
X    PageCount = 0;
X}
X
Xpageline( s )
Xchar * s;
X{
X    char c;
X
X    fputs( s, stdout );
X    if ( ++PageCount > 24 ) {
X         PageCount = 0;
X         c = get_one();
X         switch( c ) {
X         case 'q':
X         case 'Q':
X         case '\003':
X         case 'x':
X         case 'X':
X              return( TRUE );
X         }
X    }
X    return( FALSE );
X}
X
Xchar *getnext( s, p )
Xregister char **s, *p;
X{
X    *s = (char *)NULL;
X    while ( *p == ' ' || *p == '\t' )
X         p++;
X    if ( *p == '\n' || *p == '\0' )
X         return( (char *)NULL );
X    *s = p;
X    while ( *p != ' ' && *p != '\t' && *p != '\n' && *p != '\0' )
X         *p++;
X    if ( *p != '\0' )
X         *p++ = '\0';
X    return(p);
X}
SHAR_EOF
chmod 0600 mail.c || echo "restore of mail.c fails"
sed 's/^X//' << 'SHAR_EOF' > mailhost.c &&
X/*      mailhost.c
X
```

```
X*/
X#define  MAIN mailmain
X#include "host.c"
X
SHAR_EOF
chmod 0600 mailhost.c || echo "restore of mailhost.c fails"
sed 's/^X//' << 'SHAR_EOF' > makefile &&
XDESTDIR=.
X
XCFLAGS=-g
X
XUTLOBS= dirsubs.o genv.o ndir.o fgets.o getone.o\
X    inoutpth.o shell.o time.o chekname.o
XDCPOBS=  dcpgpkt.o dcpsys.o dcpxfer.o
X
XMAILOBS=mailhost.o mail.o lmail.o dmymath.o lib.o libdcp.a libutl.a
XUUOBS=uuhost.o uusup.o dcp.o rmail.o dmymath.o lib.o ndir.o libutl.a libdcp.a
X
Xall: mail.ttp uu.ttp
X
Xmail.ttp: $(MAILOBS)
X    $(CC) $(CFLAGS) -o $@ $(MAILOBS)
X
Xuu.ttp: $(UUOBS)
X    $(CC) $(CFLAGS) -o $@ $(UUOBS)
X
Xlibutl.a: $(UTLOBS)
X    c:\megamax\ar.ttp rv $@ $?
Xlibdcp.a: $(DCPOBS)
X    c:\megamax\ar.ttp rv $@ $?
X
Xrmail.o: rmail.c pcmail.c
Xlmail.o: lmail.c pcmail.c
SHAR_EOF
chmod 0600 makefile || echo "restore of makefile fails"
sed 's/^X//' << 'SHAR_EOF' > mlib.c &&
X/*      mlib.c
X
X*/
X
X#include "local/mlib.c"
X
SHAR_EOF
chmod 0600 mlib.c || echo "restore of mlib.c fails"
sed 's/^X//' << 'SHAR_EOF' > ndir.c &&
X/*
X * Program: ndir.c
X * Facility:
X *  Directory reading support for the atari st
X */
X
X#define DEF_NDIR
X
```

```
X#include <osbind.h>
X#include <stdio.h>
X#include "ndir.h"
X
X#ifndef TRUE
X#define TRUE (-1)
X#define FALSE    (0)
X#endif
X
X#define BSLASH    '\\'
X#define SLASH     '/'
X
XDIR    DtaBuff;            /* used by GEMDOS for Fsfirst/Fsnext */
Xshort  FirstSearch;        /* used to first call to readdir     */
Xstruct direct dentry;      /* portable directory structure      */
X
XDIR *opendir( filename )
Xchar *filename;
X{
X    int status;
X    char *cp1,*cp2;
X
X    Fsetdta( &DtaBuff );
X    FirstSearch = TRUE;
X
X    /*
X     * Copy filename into FilePath buffer and convert forward slashs
X     * to backward slashes.
X     * Also make sure path terminates in a back slash
X     */
X    cp1 = DtaBuff.FilePath;
X    cp2 = filename;
X    while (*cp2 != '\0')
X    {
X      *cp1++ = (*cp2 == SLASH) ? BSLASH : *cp2;
X      cp2++;
X    }
X    if ( *(cp1-1) != BSLASH )   /* enforce trailing back slash */
X    {
X      *cp1++ = BSLASH;
X    }
X    *cp1   = '\0';              /* add trailing zero */
X    strcat( DtaBuff.FilePath, "*.*" );
X
X    status = Fsfirst( DtaBuff.FilePath, 0 );
X    if ( status != 0 ) return( (DIR *)NULL );
X    return( &DtaBuff );
X}
X
Xstruct direct *readdir( dirp )
XDIR *dirp;
X{
X    int status;
```

```
X
X    if ( FirstSearch )
X    {
X      FirstSearch     = FALSE;
X      dentry.d_ino    = 0L;
X      dentry.d_reclen = 264;
X      dentry.d_namlen = strlen( dirp->FileName );
X      strcpy( dentry.d_name, dirp->FileName );
X      return( &dentry );
X    }
X    Fsetdta( dirp );
X    if (( status = Fsnext() ) != 0 )
X      return( (struct direct *)NULL );
X    dentry.d_ino    = 0L;
X    dentry.d_reclen = 264;
X    dentry.d_namlen = strlen( dirp->FileName );
X    strcpy( dentry.d_name, dirp->FileName );
X    return( &dentry );
X}
X
Xvoid closedir( dirp )
XDIR *dirp;
X{
X    return;
X}
X
X
SHAR_EOF
chmod 0600 ndir.c || echo "restore of ndir.c fails"
sed 's/^X//' << 'SHAR_EOF' > ndir.h &&
X/* @(#)ndir.h     1.4  4/16/85 */
X/* Modified for the atari st series 5/4/87 */
X#ifndef DEV_BSIZE
X#define  DEV_BSIZE 512
X#endif
X#define DIRBLKSIZ DEV_BSIZE
X#define  MAXNAMLEN 255
X
Xstruct  direct {
X    long d_ino;             /* inode number of entry */
X    short    d_reclen;      /* length of this record */
X    short    d_namlen;      /* length of string in d_name */
X    char d_name[MAXNAMLEN + 1]; /* name must be no longer than this */
X};
X
X/*
X * The DIRSIZ macro gives the minimum record length which will hold
X * the directory entry.  This requires the amount of space in struct direct
X * without the d_name field, plus enough space for the name with a terminating
X * null byte (dp->d_namlen+1), rounded up to a 4 byte boundary.
X */
X
X#ifdef DIRSIZ
```

```
X#undef DIRSIZ
X#endif
X#define DIRSIZ(dp) \
X    ((sizeof (struct direct) - (MAXNAMLEN+1)) + (((dp)->d_namlen+1 + 3) &~ 3))
X
X/*
X * Definitions for library routines operating on directories.
X */
X
X/* Structure of a directory entry under GEMDOS */
X
Xtypedef struct _dirdesc
X  {
X    char Reserved[21];
X    char FileAttr;
X    int      CreateTime;
X    int      CreateDate;
X    long FileSize;
X    char FileName[14];
X    char FilePath[256];
X  } DIR;
X
X#ifndef NULL
X#define NULL 0L
X#endif
X
X#ifndef DEF_NDIR
Xextern  DIR *opendir();
Xextern  struct direct *readdir();
Xextern  void closedir();
X#endif
X
SHAR_EOF
chmod 0600 ndir.h || echo "restore of ndir.h fails"
sed 's/^X//' << 'SHAR_EOF' > pcmail.c &&
X/*  pcmail.c
X
X
X
X    copyright (C) 1987 Stuart Lynne
X
X    Copying and use of this program are controlled by the terms of the
X    Free Software Foundations GNU Emacs General Public License.
X
X
X    version       0.1      March 31/1987
X
X
Xpcmail
X
X    pcmail address1 address2 ... < the.message
X
Xdescription
```

```
X
X    An 822 compatible (hopefully) mail delivery system for pc's.
X
X    Designed for delivering mail on a pc based system. It will put
X    local mail (ie, not @ or ! in address) into files in the default
X    mail directory.
X
X    If remote it will put into an outgoing mailbag in the default mail
X    directory. Performs a simple bundling of mail messages into one
X    file with arguments prepended as To: arg header lines. And adds a
X    Message-Lines: header which gives the number of lines in the
X    content part of the message (after the first blank line).
X
X         pcmail john jack jill@xyz.uucp < afile
X
X         To: john
X         To: jack
X         To: jill@xyz.uucp
X         X-Message-Lines: ?????
X         Content-Length: ?????
X
X         ...
X         ...
X         ...
X
X    Content-Length: is used without X- prepended to be compatible with AT&T
X    Mail bundles. This is not 822 compatible per se, but is allowed.
X
X    It also adds the from From and Date lines. Subject: lines may be inserted
X    by placing them at the beginning of the message.
X
X    A Unix version should lock the /usr/mail/mailbag file.
X
X    Another program called rpcmail will unbundle the files created by
X    pcmail and deliver each message to the local rmail. So conceptually
X
X         (pcmail ..... < ...; pcmail .... < ...) | sz -> rz | rpcmail
X
X    would deliver remote messages intact.
X
Xenvironment variables
X
X    The following evironment variables are used:
X
X         MAILBOX       current user's mailbox,     "stuart"
X         NAME     current user's name,        "Stuart Lynne"
X         DOMAIN        domain of this machine,     "slynne.mac.van-bc.can"
X         MAILDIR       where is mail kept,         "mpw:mail"
X
Xcompiling
X
X    Compiled by itself it will operate as a standalone program. If the
X    compiler option:
```

```
X
X        -DNOMAIN
X
X    is used, it will compile as a routine:
X
X        pcmail (argc, argv)
X        char **argv;
X        int argc;
X
X    and can be used internally in other programs.
X
X
XCustomization
X
X    PCMAIL        mailbag for remote mail
X    FAKEUUX       emulate uux, make appropriate files in SPOOLDIR
X
X    RMAIL         rmail
X
X    DEBUG1        first level Debug trace
X
X
X*/
X
X#include "dcp.h"
X
XFILE *FOPEN();
X
X#define FORWARD       "Forward to"
X
X
X#define  SBUFSIZ       124
X
XFILE   *mailfile;
XFILE   *tempfile;
X
Xchar    buf[BUFSIZ];
Xchar    miscbuff[100];
Xlong int lines = 0;
Xlong int bytes = 0;
Xlong int sequence = 0;
X
Xlong tloc;
Xchar chartime[26];    /* current time in characters */
Xchar *thetime;
X
Xchar    tfilename[100];
Xchar    mfilename[100];
Xchar    mailsent[100];
Xchar    s_systems[64];
X
Xint local = TRUE;
X
```

```
Xchar    remotes[BUFSIZ];
X
X
Xchar uucp[] = "uucp";
X
Xchar *fgets();
Xint fputs();
X
X#ifdef NOMAIN
X
X#ifdef RMAIL
X#define main rmail
X#else
X#define  main lmail
X#endif
X
X#define exit return
X
Xextern int debuglevel;
X
X#else
Xint debuglevel;
X#endif
X
X#ifndef RMAIL
Xchar    Subject[132] = "";
X#endif
X
Xchar *mcurdir;
Xchar s_mcurdir[128];  /* current directory path (save malloc call) */
Xchar * getcwd();
Xint chdir();
X
X
X
Xmain(argc, argv)
Xchar *argv[];
X{
X    long int position;
X    register int header = 1;
X    register int amount;
X
X    register int argcount;
X    register char **argvec;
X    int remote;
X    int s1, s2;
X
X
X
X
X#ifndef NOMAIN
X    /* get environment var's */
X    HOSTINIT;
```

```
X    loadenv();
X
X    if (argc <= 1) {
X         fprintf( stderr, "pcmail usage: pcmail addresses < message\n" );
X         exit(1);
X    }
X    debuglevel = 99;
X
X#endif
X#ifdef RMAIL
X    local = FALSE;
X#else
X    local = TRUE;
X#endif
X
X    if ( debuglevel > 5 ) {
X         fprintf( stderr, "pcmail: argc %d ", argc );
X         argcount = argc;
X         argvec = argv;
X         while (argcount--) {
X              fprintf( stderr, " \"%s\"", *argvec++ );
X              }
X         fprintf( stderr, "\n" );
X
X         tloc = time( (long *)NULL );
X         thetime = ctime(&tloc);
X         fprintf( stderr, "thetime: %s\n",thetime );
X    }
X
X#ifdef MSDOS
X    mcurdir = getcwd( s_mcurdir, sizeof(s_mcurdir) );
X#else
X    mcurdir = getcwd( s_mcurdir, 0 );
X#endif
X    chdir( spooldir );
X
X    /* get sequence number */
X    mkfilename( tfilename, confdir, SFILENAME );
X
X    if ( debuglevel > 4 )
X       fprintf( stderr, "pcmail: opening %s\n", tfilename ); /* */
X
X    tempfile = FOPEN( tfilename, "r", 't' );
X    if (tempfile != (FILE *)NULL) {
X         fscanf( tempfile, "%ld", &sequence );
X         fclose( tempfile );
X    }
X    else {
X         fprintf( stderr, "pcmail: can't find %s file, creating\n",
X                        tfilename );
X         sequence = 1; /* start at 1 */
X    };
X
```

```
X    /* update sequence number */
X    if ( debuglevel > 5 )
X         fprintf( stderr, "pcmail: new sequence # %ld\n", sequence );
X
X    tempfile = FOPEN( tfilename, "w", 't' );
X    if (tempfile != (FILE *)NULL) {
X         fprintf( tempfile, "%ld\n", sequence+1 );
X         fclose( tempfile );
X    }
X
X    /* open a temporary file */
X    /* sprintf( tfilename, TFILENAME, sequence ); */
X    sprintf( miscbuff, TFILENAME, sequence );
X    mkfilename( tfilename, tempdir, miscbuff );
X
X    if ( debuglevel > 5 )
X         fprintf( stderr, "pcmail: opening %s\n", tfilename );
X
X    tempfile = FOPEN( tfilename, "w", 'b' );
X    if (tempfile == (FILE *)NULL)  {
X         fprintf( stderr, "pcmail: can't open %s\n", tfilename );
X         exit(1);
X    }
X
X    /* copy stdin to tempfile, counting content lines and bytes */
X    header = 1;
X    while (fgets( buf, 512, stdin ) != (char *)NULL) {
X         if (header != 0) {
X              if (strlen( buf ) == 1) {
X                   header = 0;
X                   fprintf( tempfile, "\n" );
X                   continue;
X              }
X              else if (strchr( buf, ':' ) == NULL) {
X                   header = 0;
X                   fprintf( tempfile, "\n" );
X              }
X         }
X         if (header == 0) {
X              lines++;
X              bytes += strlen( buf );
X         }
X         fputs( buf, tempfile );
X    }
X
X    fclose( tempfile );
X
X    if ( debuglevel > 4 ) {
X         fprintf( stderr, "pcmail: stdin copied to tmp %ld %ld\n",
X                        bytes, lines );
X         fprintf( stderr, "pcmail: args %d\n", argc );
X    }
X
```

```
X    /* loop on args, copying to appropriate postbox,
X       do remote only once
X       remote checking is done empirically, could be better
X    */
X    remotes[0] = '\0';
X
X
X#ifndef RMAIL
X    if ( strcmp( argv[1], "-s" ) == SAME ) {
X         argv++;argv++;
X         argc--;argc--;
X         if ( argc == 0 )
X              return( -1 );
X         strcpy( Subject, *argv );
X         }
X#endif
X    argcount = argc;
X    argvec = argv;
X
X    while (--argcount > 0) {
X         argvec++;
X         if ( debuglevel > 5 )
X              fprintf( stderr, "pcmail: arg# %d\ %s\n",
X                                argcount, *argvec );
X
X         if (
X              (strchr( *argvec, '!' ) != SAME) ||
X              (strchr( *argvec, '@' ) != SAME) ||
X              (strchr( *argvec, '%' ) != SAME)
X            ) {
X              if ( debuglevel > 5 )
X                   fprintf( stderr, "pcmail: send to remote\n" );
X
X              s1 = strlen( remotes );
X              s2 = strlen( *argvec );
X
X              /* can we cram one more address on line */
X              if ( s1 > 0 && (s1 + s2 + 1) > 128 ) {
X                   /* dump it then, to bad */
X                   sendone( argc, argv, remotes, TRUE );
X                   remotes[0] = '\0';
X              }
X
X              /* add *arvgvec to list of remotes */
X              strcat( remotes, " " );
X              strcat( remotes, *argvec );
X
X         }
X         else {
X              if ( debuglevel > 5 )
X                   fprintf( stderr, "pcmail: calling sendone %s\n",
X                                 *argvec );
X
```

```
X             sendone( argc, argv, *argvec, FALSE );
X        }
X
X    }
X    /* dump remotes if necessary */
X    if ( strlen( remotes ) > 0 )
X         sendone( argc, argv, remotes, TRUE );
X
X#ifndef RMAIL
X
X    mkfilename( mailsent, home, COPYFILE );
X    if ( debuglevel > 4 )
X       fprintf( stderr, "pcmail: copfile = %s\n", mailsent );
X    sendone( argc, argv, mailsent, FALSE );
X
X#endif
X
X    unlink( tfilename );
X    chdir( mcurdir );
X    exit(0);
X}
X
X/*
X * Figure out which mail server we should send something to.  Scan the
X * systems file for the first component of the desired mailpath.  If we
X * don't find it, revert to the default server.
X */
Xservername(address, def, buf)
Xchar *address, *def, *buf;
X{
X    char temp[32], line[256], *bang;
X
X    while (*address == ' ')
X         address++;
X    strncpy(temp, address, 31);
X    temp[31] = '\0';
X
X/* Isolate the first component of the mail path */
X    bang = index(temp, '!');
X    if (bang)
X         *bang = '\0';
X
X    if (checkname(temp) == FAILED)
X         strcpy(buf, def);
X    else
X         strcpy(buf, temp);
X}
X
X
Xchar fpat1[] = "%c.%.7s%04ld";
Xchar fpat2[] = "S %s %s %s - %s 0666 %s";
X
X/* sendone copies file plus headers to appropriate postbox
```

```
X   NB. we do headers here to allow flexibility later, for example
X   in being able to do bcc, per host service processing etc.
X*/
Xsendone( argc, argv, address, remote )
Xchar **argv;
Xint argc;
Xchar *address;
X{
X#ifdef RMAIL
X    int  recflag = 0;       /* we have read a "Received:" */
X    int  fromflag = 0;      /* we have copied the "From " */
X#endif
X    register char      *cp;
X    char maildest[32];      /* system to send to */
X    char icfilename[32];         /* local C. copy file */
X    char ixfilename[32];         /* local X. xqt file */
X    char idfilename[32];         /* local D. data file */
X    char rxfilename[32];         /* remote X. xqt file */
X    char rdfilename[32];         /* remote D. data file */
X    char tmfilename[32];         /* temporary storage */
X#ifdef MSDOS
X    char cixfilename[32];        /* canonical ixfilename */
X    char cidfilename[32];        /* canonical idfilename */
X#endif
X    if ( remote ) {
X         /* sprintf all required file names */
X         mkfilename(s_systems, confdir, SYSTEMS);
X         servername(address, mailserv, maildest);
X         sprintf( tmfilename, fpat1, 'C', maildest, sequence );
X         importpath( icfilename, tmfilename );
X#ifdef MSDOS
X         sprintf( cidfilename, fpat1, 'D', maildest, sequence );
X         importpath( idfilename, cidfilename );
X         sprintf( cixfilename, fpat1, 'D', nodename, sequence );
X         importpath( ixfilename, cixfilename );
X#else
X         sprintf( tmfilename, fpat1, 'D', maildest, sequence );
X         importpath( idfilename, tmfilename );
X         sprintf( tmfilename, fpat1, 'D', nodename, sequence );
X         importpath( ixfilename, tmfilename );
X#endif
X         sprintf( rdfilename, fpat1, 'D', nodename, sequence );
X         sprintf( rxfilename, fpat1, 'X', nodename, sequence );
X    }
X    else {
X         /* postbox file name */
X         if ( index( address, SEPCHAR ) == (char *)NULL )
X              mkfilename( idfilename, maildir, address );
X         else
X              strcpy( idfilename, address );
X    }
X
X    if ( debuglevel > 5 )
```

```
X         fprintf( stderr, "pcmail: sendone: %s\n", idfilename );
X
X    if ( remote == FALSE ) {
X       if ( debuglevel > 5 )
X          fprintf( stderr, "pcmail: sendone: check for remote\n" );
X         /* check for forwarding */
X         if ( (mailfile = FOPEN( idfilename, "r", 'b' )) != (FILE *)NULL ) {
X              cp = fgets( buf, BUFSIZ, mailfile );
X              fclose( mailfile );
X              if (cp != (char *)NULL)
X                   if (strncmp( buf, FORWARD, 10 ) == 0) {
X                        strcpy( buf, buf+11 );
X                        return( sendone( argc, argv, buf, FALSE ) );
X                   }
X         }
X    }
X
X    /* open mailfile */
X    if ( (mailfile = FOPEN( idfilename, "a", 'b' )) == (FILE *)NULL ) {
X         fprintf( stdout, "pcmail: cannot append to %s\n", idfilename );
X         return( 0 );
X    }
X
X    if ( debuglevel > 5 )
X         fprintf( stderr, "pcmail: append to mailfile\n" );
X
X    tloc = time( (long *)NULL );
X    thetime = ctime(&tloc);
X    (void)strcpy(chartime, thetime); /* make our own copy */
X    thetime = chartime;    /* and work with our own copy */
X    thetime[strlen(thetime)-1] = '\0';
X
X#ifndef RMAIL
X    fprintf( mailfile, "From %s %s", mailbox, thetime );
X    if ( remote )
X         fprintf( mailfile, " remote from %s", nodename );
X    fputc( '\012', mailfile );
X    fprintf( mailfile, "Received: by %s (pcmail) %s", domain, thetime );
X    fputc( '\012', mailfile );
X    fprintf( mailfile, "Date: %s", thetime );
X    fputc( '\012', mailfile );
X    /* add Date:, From: and Message-ID: headers */
X    fprintf( mailfile, "From: %s <%s@%s>", name, mailbox, domain );
X    fputc( '\012', mailfile );
X    fprintf( mailfile, "Message-ID: <%ld@%s>", sequence, domain );
X    fputc( '\012', mailfile );
X    /* add To: headers */
X    while (--argc > 0) {
X         fprintf( mailfile, "To: %s", *++argv );
X         fputc( '\012', mailfile );
X    }
X    if ( strlen( Subject ) > 0 ) {
X         fprintf( mailfile, "Subject: %s", Subject );
```

```
X        fputc( '\012', mailfile );
X    }
X#ifdef PCMAIL
X    /* add Message-Lines: and Content-Length: headers */
X    fprintf( mailfile, "X-Message-Lines: %ld", lines );
X    fputc( '\012', mailfile );
X    fprintf( mailfile, "Content-Length: %ld", bytes );
X    fputc( '\012', mailfile );
X#endif /* PCMAIL */
X#endif /* RMAIL */
X
X    /* copy tempfile to postbox file */
X    if (debuglevel > 4)
X       fprintf( stderr, "pcmail: copy tempfile %s to %s\n",
X                        tfilename, idfilename );
X    tempfile = FOPEN( tfilename, "r", 'b' );
X    if ( tempfile == (FILE *)NULL) {
X        fprintf( stdout, "pcmail: can't re-open %s\n", tfilename );
X        return( 0 );
X    }
X    while (fgets( buf, 512, tempfile ) != (char *)NULL) {
X/*
X * If this is rmail, we want to munge the first "From " line, since there
X * will be a "remote from hostname" at the end that we want to prepend to
X * the mail path after "From ".  Subsequent "From " lines will have a >
X * prepended so we don't mess up the mail shell (all "From " lines are
X * treated this way if we aren't rmail)
X */
X        if ( strncmp( buf, "From ", 5 ) == 0 )
X#ifdef RMAIL
X             if (! fromflag++)
X                  munge_from(buf);
X             else
X#endif
X                  fputc( '>', mailfile );
X/*
X * We want to insert our "Received:" line above all the others, if this
X * is rmail.  There will be some if we're receiving from a remote site
X * (i.e., rmail) -- if we aren't, the received line is inserted elsewhere
X * anyway so we don't worry about it.
X */
X#ifdef RMAIL
X        if ((! strncmp(buf, "Received:", 9)) && (! recflag++))
X             fprintf( mailfile, "Received: by %s (pcmail) %s\n",
X                  domain, thetime );
X#endif
X        cp = &buf[ strlen(buf)-1 ];
X        if ( *cp == '\n' )
X             *cp = '\0';
X        fputs( buf, mailfile );
X        fputc( '\012', mailfile );
X    }
X
```

```
X    /* close files */
X    fclose( mailfile );
X    fclose( tempfile );
X
X    /* all done unless going to remote via uucp */
X    /* must create the job control files */
X    if ( remote == TRUE ) {
X
X         /* create remote X xqt file */
X         mailfile = FOPEN( ixfilename, "w", 'b' );
X         if   (mailfile == (FILE *)NULL) {
X              fprintf( stdout, "pcmail: cannot append to %s\n", ixfilename );
X              return( 0 );
X         }
X         fprintf( mailfile, "U %s %s", uucp, nodename );
X         fputc( '\012', mailfile );
X         fprintf( mailfile, "F %s", rdfilename );
X         fputc( '\012', mailfile );
X         fprintf( mailfile, "I %s", rdfilename );
X         fputc( '\012', mailfile );
X         fprintf( mailfile, "C rmail %s", address );
X         fputc( '\012', mailfile );
X         fclose( mailfile );
X
X         /* create local C copy file */
X         mailfile = FOPEN( icfilename, "w", 't' );
X         if   (mailfile == (FILE *)NULL) {
X              fprintf( stdout, "pcmail: cannot append to %s\n", icfilename );
X              return( 0 );
X         }
X
X#ifdef MSDOS
X         fprintf( mailfile, fpat2, cidfilename, rdfilename,
X                        uucp, cidfilename, uucp );
X         fputc( '\012', mailfile );
X         fprintf( mailfile, fpat2, cixfilename, rxfilename,
X                        uucp, cixfilename, uucp );
X         fputc( '\012', mailfile );
X#else
X         fprintf( mailfile, fpat2, idfilename, rdfilename,
X                        uucp, idfilename, uucp );
X         fputc( '\012', mailfile );
X         fprintf( mailfile, fpat2, ixfilename, rxfilename,
X                        uucp, ixfilename, uucp );
X         fputc( '\012', mailfile );
X#endif
X         fclose( mailfile );
X
X    } /* if ( remote == TRUE ) */
X
X    return( 1 );
X}
X
```

```
X/*
X * Munge a "From " line.  If there is a " remote from hostname" at the
X * end of the line, prepend hostname to whatever comes after the "From "
X * and get rid of " remote...".  Otherwise leave the line alone.  Since
X * we will be doing only this munge, the resulting From line will never
X * be longer than the original, so copying back into the original's
X * buffer is okay.
X */
Xmunge_from(buf)
Xchar *buf;
X{
X    int rstart, plen;
X    char path[256], date[256], host[16];
X
X    if (sscanf(buf, "From %s ", path) != 1)
X        return;
X
X    plen = strlen(path);
X
X    for (rstart = plen+6; buf[rstart]; rstart++)
X        if (! strncmp(buf+rstart, " remote from ", 13))
X            break;
X
X    if (! buf[rstart])
X        return;
X
X    buf[rstart] = 0;
X    strcpy(date, buf+plen+6);
X
X    strcpy(host, buf+rstart+13);
X    plen = strlen(host);
X    if (! plen)
X        return;
X
X    if (host[plen-1] == '\n')
X        host[plen-1] = 0;
X
X    sprintf(buf, "From %s!%s%s\n", host, path, date);
X}
X
X
X#ifndef AMIGA
X#ifdef RMAIL
Xrnews(argc, argv)
Xint argc;
Xchar *argv[];
X{
X    struct tm *thetm;
X    char filename[132];
X    char format[128];
X    FILE     *f;
X    char buf[BUFSIZ];
X
```

```
X    static int count = 0;
X
X    tloc = time( (long *)NULL );
X    thetime = ctime(&tloc);
X    tloc = time( (long *)NULL );
X
X    thetm = localtime( &tloc );
X
X/*
X * Construct a filename to spool news to temporarily.  The filename looks
X * like DDHHMMSS.CCC, where DD is the day of the month, HH is the hour,
X * MM is the minute, SS is the second, and CCC is an increasing sequence
X * counter.
X */
X    sprintf( filename, NEWSDIR,
X         thetm->tm_mday,
X         thetm->tm_hour,
X         thetm->tm_min,
X         thetm->tm_sec,
X         count
X         );
X
X    count++;
X
X    if ( debuglevel > 5 )
X         fprintf( stderr, "rnews: %s\n", filename );
X
X    if ( (f = FOPEN( filename, "w", 't' )) == (FILE *)NULL ) {
X         fprintf( stderr, "rnews: can't open %s %d\n", filename, errno );
X         return( -1 );
X    }
X
X    while ( fgets( buf, BUFSIZ, stdin ) != (char *)NULL )
X         fputs( buf, f );
X
X    fclose( f );
X}
X
X#endif /* RMAIL */
X#endif /* AMIGA */
X
X
SHAR_EOF
chmod 0600 pcmail.c || echo "restore of pcmail.c fails"
sed 's/^X//' << 'SHAR_EOF' > printmsg.c &&
X#include <stdio.h>
X#include <strings.h>
Xextern FILE *logfile;
Xextern int debuglevel;
Xextern int remote;
X
X#define MASTER 1
X
```

```
X/* */
X/*
X *  p r i n t m s g
X *
X *  Print error message on standard output if not remote.
X */
X/*VARARGS1*/
Xprintmsg(level, fmt, a1, a2, a3, a4, a5)
Xint     level;
Xchar    *fmt;
Xchar    *a1, *a2, *a3, *a4, *a5;
X{
X    char msg[256];
X
X    if ( debuglevel > level ) {
X         sprintf( msg, fmt, a1, a2, a3, a4, a5 );
X         strcat( msg, "\n" );
X         if ( remote == MASTER )
X              fputs( msg, stdout );
X         fputs( msg, logfile );
X    }
X}
X
X
SHAR_EOF
chmod 0600 printmsg.c || echo "restore of printmsg.c fails"
sed 's/^X//' << 'SHAR_EOF' > readme.1st &&
XUUPC/ST version 0.1 "release" notes
X
X
X                              *** CAVEATS **
X
X        I'm releasing this very unfinished version of UUPC because I've
Xgotten so many requests for it.  I make no claims about the cleanliness
Xor functionality of this code; it is by no means debugged and is very
Xrough in a number of places.  In particular:
X
X- The program only works in originate mode, not answer mode.  This means
X  that your ST can poll other systems, but they can't poll you.
X
X- Newsfeeds are only minimally supported.  The program will take non-batched,
X  non-compressed articles as input to the "rnews" program and put them in a
X  spool directory, by default \usr\spool\rnews, with filenames indicating
X  the date and time that the articles were received.  No attempt is made to
X  sort articles by newsgroup.
X
X- The mail shell is primitive, to say the least.  I have improved on the
X  functionality of the shell in the original uupc distribution, but it
X  still lacks many features.
X
X- Configuration requires modification of a number of source files, which
X  will be listed later, and knowledge of some UUCP file formats (L.sys).
X
```

X	All of the above, and many more bugs and deficiencies, will be
Xfixed in the (near?) future. I stress again that I'm releasing this
Xsource code only because lots of people wanted to see it, not because I
Xthink it's anywhere near ready to distribute.
X
X	This code was developed with Laser C; it may or may not compile
Xor work under other systems.
X
X	I can't take credit for writing most of the code; read the other
Xdocumentation files for its history. I downloaded UUPC from uunet and
Xmodified it from there.
X
X
X
X			*** CONFIGURATION ***
X
X	Several files will have to be modified to configure the program
Xfor your site. If you can't get it working, too bad. That sounds harsh,
Xbut I have neither the time nor the inclination to answer 20 mail messages
Xa day about code that isn't finished yet. More complete documentation will
Xbe included with the final release of the code
X
XGENV.H Most of the items in this file should be self-explanatory; I
X	have included the defaults from my site.
X
XHOST.H The primary items of interest here are COPYFILE (which keeps a
X	copy of outgoing mail) and NEWSDIR, which is the directory to
X	stash news articles in. COPYFILE is relative to the home
X	directory specified in GENV.H.
X
XSYSTEMS This is UUPC's L.sys file; it lives in CONFDIR. I've included my
X	systems file, sans phone numbers and passwords (don't want to
X	upset my feeds!). Note that it differs somewhat from the standard
X	L.sys format; in particular, the "speed" field is the Hayes command
X	(without AT) to dial out to the target system.
X
X
X	I *think* those cover the configuration; the final release will
Xautomate configuration, maybe with a GEM shell (but I'm not promising
Xanything). I am also porting netnews 3.0 to the ST, so the final release
Xwill have a completely functional news system.
X	This code should be modified to run cleanly under MT C-shell;
Xit will hog CPU time like crazy as is. I plan to reorganize the code
Xto make porting to other environments easier.
X
X	If you have bug fixes or improvements, please mail them to me.
XI'd also like bug reports from people who get the system to compile, but
Xplease don't bother me with things like "it doesn't let you edit mail."
XSuggestions for new features will have to wait until I'm done with the
Xstuff *I* want to see -- I probably have 90% of the suggestions covered
Xalready. Questions about how to set up the program are subject to
Xdeletion without a reply; see the rationale above. Praise and offers
Xof wealth and power will be gladly accepted, however. :]

```
X        The code really isn't as bad as I've tried to make it sound; it
Xhas been working reliably for me for the past couple of weeks, and only
Xcrashes if the host you're connected to does something really weird.
X
X
X        -Steven Grimm
X         Domain: koreth@ssyx.ucsc.edu
X         UUCP:   ...!uunet!ucbvax!ucscc!ssyx!koreth
X                              or (faster):
X                ...!uunet!ssyx.ucsc.edu!koreth
X
SHAR_EOF
chmod 0600 readme.1st || echo "restore of readme.1st fails"
sed 's/^X//' << 'SHAR_EOF' > readme.dcp &&
Xuupc/dcp              June 8, 1987      Stuart Lynne
X
XFor Beta implementors only.
X
XThis is the very first release of my version of Richard H. Lambs uucp
Xprogram dcp.
X
XSee README.UUPC for overview of how everything is stitched together.
X
XSummary of Changes:
X
X    - eliminated references to all protocols except for the 3 window
X      g protocol
X    - streamlined dcp and condensed into four files
X    - moved host dependant stuff to one file uuXXX
X    - bug fixes to get 3w g protocol to send at full speed
X
X
Xdcp.c        - dcp pseudo main, high level stuff
Xdcp.h        - header file
Xdcpxfer.c    - file xfer
Xdcpsys.c - connection stuff
Xdcpgpkt.c    - g packet protocol
X
X
XFor more information, bug fixes, more commands etc:
X
X        Stuart.Lynne@van-bc.uucp
X        604-937-7532
X
X
XFixes
X
XJun 8/87 Added FOPEN and CREAT to recursively create directory
X             trees if they don't exist when a file is opened for writing
X             or appending
X
XJun 8/87 Found bug in getting file name in initial protocol sequence
X             Need to read more than one packet.
```

```
X
XJul/87       Added Hayes dialer to dcpsys
SHAR_EOF
chmod 0600 readme.dcp || echo "restore of readme.dcp fails"
sed 's/^X//' << 'SHAR_EOF' > readme.inf &&
XAugust 9, 1987    uupc Questions and Answers     uupc Development
X
X
XThe following is some commonly asked questions about uupc and, of
Xcourse, the answers to these questions.
X
X
X 1. What does "uupc" stands for?
X
X    It is an acronym for "UUcp for PC's", but it is also a pun on
X    uucp, which is in turn an acronym for "Unix to Unix CoPy".
X
X 2. What does uupc do?
X
X    It gives a personal computer the capability to become a
X    "node" in the UUCP (or a similar) network and exchange
X    information such as electronic mail and USENET news with
X    other computers on that network.
X
X 3. What personal computers does uupc runs on?
X
X    Currently it is available for the Apple Macintosh, Atari ST,
X    Commodore Amiga, and IBM PC (and compatibles) with DOS.  More
X    computers and operating systems will be able run uupc in the
X    near future.  (IBM PC with MINIX is a likely next candidate.)
X
X 4. Does uupc require me to leave my computer on all day to wait
X    for incoming mail?
X
X    No.  Most people only use uupc to call up their neighbouring
X    system to send and/or pickup mail at times convenient to
X    them.  Outgoing mail are also spooled to disk and do not need
X    to be send immediately to your neighbouring system after it
X    is composed.
X
X    However, uupc can also be set up on a personal computer to
X    wait for incoming call continuously and act as a "mail-hub"
X    to relay messages for other systems if you choose.
X
X 5. What do I need to have to get uupc up and running on one of
X    the above personal computers?
X
X    You need a neighbouring system to communicate with.  This
X    system can be either a UNIX system, another personal computer
X    running uupc, or any other system that can talk UUCP's 'g'
X    protocol.
X
X    You would also need to have the appropriate C compiler for
```

X   your personal computer if you have received only the source
X   for uupc.
X
X 6. Is the source to uupc publicly available?
X
X   Yes. It was posted to the USENET newsgroup comp.sources.misc
X   in August 1987 and is available from (at least) any site
X   which archives this newsgroup. If you have trouble locating
X   a copy of the uupc sources, please drop uupc Development a
X   note through one of the e-mail addresses listed at the end of
X   this file.
X
X 7. What does the uupc software consists of?
X
X   It consists of two programs, uupc and pcmail. uupc is an automated
X   files transfer program, similar to /usr/lib/uucico in UUCP,
X   and mail is a mailer user-interface, like mail(1) in UNIX.
X
X 8. What are the typical use of these programs?
X
X   uu is used to accept incoming file relayed to you through
X   your neighbouring machine and deliver outgoing file to your
X   neighbouring machine for forwarding to other machines. In
X   most cases these "files" contain electronic messages which
X   are to be used with the mail program.
X
X   pcmail is used to read incoming mail delivered by uu, and
X   compose outgoing mail for delivery with uu. However, it can
X   also be used to transfer files to/from other systems that is
X   reachable through electronic mail.
X
X 9. What do I need to do to get uupc running on my personal
X   computer?
X
X   You would need to obtain the binaries of uupc for your
X   computer by either compiling the uupc sources on your machine
X   or obtaining the uupc binaries from someone who has a copy.
X
X   You would also need to arrange to have your neighbouring
X   system to recognize your system as one of their neighbouring
X   systems in the network. The procedures for this varies, you
X   should contact the people who manage your neighbouring system
X   for about details.
X
X10. Does uupc supports more than one neighbouring systems?
X
X   Yes, it can support multiple neighbouring systems. The mail
X   software will currently always route outgoing mail through
X   one of these systems, but a future version of this software
X   will allow multiple forwarding machines for outgoing mail.
X
X11. Is uupc the same program on all systems it runs on, or is it
X   actually a different program for each of the systems?

X
X   It is the same program across all systems, with the exception
X   of the system-dependent code, which is different from system
X   to system.
X
X   The user-interface and command line options for uupc are also
X   uniform across all the systems it runs on, so there is no
X   need to learn a new program when you use uupc on a different
X   computer. The uniform user-interface also makes it easier to
X   use uupc on different computers at the same time.
X
X12. If I don't like the mail program's simple user-interface, are
X   there any alternatives?
X
X   Since a mailbox can be easily converted to a simple text
X   file, alternative mailer can be easily written to accomodate
X   different needs. At the very least, you will be able to use
X   your favorite text-editor to read your incoming message and
X   compose your outgoing message.
X
X   Future release of uupc will include mailers for the different
X   systems which will take advantage of special features only
X   availabe on the systems they run on (e.g. window and mouse).
X
X13. What if I want to port uupc to another personal computer not
X   presently support by uupc?
X
X   First you should read the file UUPORT.INF, which should be
X   available from the same source you obtained this file from.
X   If you cannot locate a copy of this file, then please send a
X   request for it to uucp Development at one of the e-mail
X   addresses listed at the end of this file.
X
X   After you have read the above file and decided that you still
X   want to do a port of uupc to a new machine/operating systems,
X   please drop uupc Development a note at one of the the e-mail
X   addresses listed at the end of this file. This way we will
X   at least be able to save each other from duplicated efforts.
X   Who knows? We might even have a version for ready for your
X   system when you call to tell us that you are about to begin
X   your port.
X
X14. Who/what is the "UUPC Development Team"?
X
X   The original software (dcp) was done by Richard H. Lamb.
X   Modified to run on the Mac by Stuart Lynne.
X   Atari ST by Lawrence Harris.
X   Amiga by Jeff Lydiatt.
X   IBM PC by Samual Lam
X   VMS (not available yet) Lawrence Harris
X
X15. What is the copyright status and distribution policy of uupc?
X

X   The dcp portions of uupc are Copyright (c) Richard H. Lamb.
X   Modifications Copyright (c) Stuart Lynne
X   Mail, PCMail Copyright (c) Stuart Lynne
X   Mac software Copyright (c) Stuart Lynne
X   Amiga software Copyright (c) Jeff Lydiatt
X   Atari software Copyright (c) Lawrence Harris
X   IBM software Copyright (c) Sam Lam
X
XIn general we are promoting the use of this software on a "public domain"
Xbasis. You can use for your own use, and can give copies of the source
Xcode to anyone, provided you provide this information to them.
X
X
X16. If I have more questions, comments, or suggestions about
X   uupc, where should I send them?
X
X   Please send them all to us at uupc Development at one of the
X   e-mail addresses listed below. We also welcome any bug fixes
X   and improved/new code for uupc that you might want to share.
X
X
Xuupc Development can be reached at the following e-mail address:
X
X   uupc@van-bc.UUCP
X
XThis is routed to the uupc mailing list and a local news group for
Xdiscussion of uupc software.
X
XTo join the mailing list send a request to:
X
X   uupc-request@van-bvc.uucp
X
X
X
X17. Can I get Binary Versions of uupc mailed to me.
X
XYes and no.
X
XNo we cannot email binaries to you at this time.
X
XYes, if you send a self addressed / stamped (international coupon) mailer
Xwith appropriate diskettes (2) we will attempt to return them to you with
Xthe appropriate version of the software.
X
XWe plan to make a binary posting to the appropriate Usenet comp.binary
Xnewsgroups in the late fall, or early next year when the software is
Xa bit more functional, better documented and easier to install and
Xoperate without the source.
X
XMail your disks to:
X
X   UUPC Request
X   C/O Stuart Lynne

```
X    225B Evergreen Drive
X    Port Moody, BC,
X    Canada, V3H 1S1
X
X
X--
X{ihnp4!alberta!ubc-vision,uunet}!van-bc!Stuart.Lynne Vancouver,BC,604-937-7532
X
SHAR_EOF
chmod 0600 readme.inf || echo "restore of readme.inf fails"
sed 's/^X//' << 'SHAR_EOF' > readme.por &&
Xuupc Porting Information     August 3, 1987      uupc Development
X
X
XThis writeup is aimed at programmers intending to port uupc to
Xmachines and operating systems which it don't currently run on.
XIf you only want to install uupc on a system where a uupc port
Xexists, this writeup might also provide some insights to the
Xinternal working of uupc and make the installation of uupc on
Xyour machine a more interesting task.
X
XSince I did part of the IBM-PC/MS-DOS port for uupc, what I will
Xdescribe here is actually closely related to the structure of
Xthis particular port.  (I also did part of the port of uupc's
Xmailer to AIX (SVR1) on the RT-PC.)  What is shown here is *by no
Xmean* the only way things could be done, it's just some tips and
Xhints that I can think of after doing one-and-a-half port of
Xuupc.
X
XThe purpose of this writeup is to try to make the life of
Xprogrammers doing uupc ports for other machines/operating systems
Xeasier.  However, the best sources of information on how new
Xports should be done is still by reading the system-dependent
Xcode of the existing uupc ports.  (This is how I learnt how to do
Xmy ports.)  It would also be useful to have a listing of the
Xsystem-dependent code of the IBM-PC/DOS port handy while reading
Xthis writeup.
X
X
XTo compile uupc, the C compiler and linker on your system must be
Xable to differentiate between upper and lower cases in external
Xnames.
X
XThe C run-time support on your system should use '\n' as line
Xseparator for *text* files.  If your system uses other sequences
Xor methods to delimit text file lines, the C run-time support
Xmust perform the appropriate translation functions for mapping in
Xboth directions when a file is opened in text mode.
X
XIn order to port uupc to a new machine/operating system, you have
Xto come up with a new version of six system-dependent files in
Xthe the "local" directory.  The names of these six files in the
Xlocal directory are host.h, host.c, mlib.c, ulib.c, ndir.h and
```

```
Xndir.c.
X
XNote that theses files could in turn #include other *.c and *.h
Xfiles if you wish to separate the system-dependent code into more
Xsmall files.  The important thing here is that they must
Xcollectively provide the same set of routines, global variables,
Xand environment to the common code.
X
XWhat should be contained in these six files are describe below:
X
X
Xhost.h - Host dependent file virtual #included everywhere else.
X
X   Almost every file in both the common and host-dependent code
X   include this header file.  So it should contains all the
X   #includes, #defines, and externs that everybody would need.
X
X   Parts of the uupc common code assume Berkeley-style index()
X   and rindex(), so if your system supports only SysV-style
X   strchr() and strrchr(), they need to mapped using #define
X   here.
X
X   Declarations for "library" routines in host.c, mlib.c and
X   ulib.c should also be make here to made them known to the rest
X   of the world.  However, declaractions for the directory
X   scanning routines should be put into ndir.h instead.
X
X   If your system requires that text file and binary fiies be
X   opened differently, you should map the name 'FILEMODE' into
X   'filemode' with a #define here.  Otherwise, 'FILEMODE' should
X   be mapped to the null string.
X
X
Xhost.c - Generic main program and library routines for both parts.
X
X   This file includes a generic main program that simply starts
X   up and call the procedure MAIN.  More importantly, the
X   definition of all the library routines that are needed by both
SHAR_EOF
echo "End of part 3, continue with part 4"
echo "4" > s2_seq_.tmp
exit 0


From -5679271666382964908
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,1142a0cbd8424376,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-07-29 10:10:00 PST
Path: utzoo!utgpu!water!watmath!clyde!att!osu-cis!tut.cis.ohio-state.edu!cwjcc!
gatech!rutgers!mailrus!uwmcsd1!ig!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
```

Subject: v01i068: uupc -- UUCP clone (pre-pre-release) part04/04
Keywords: Laser, buggy
Message-ID: <4335@saturn.ucsc.edu>
Date: 29 Jul 88 17:10:00 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1197
Approved: koreth@ssyx.ucsc.edu

Submitted-by: koreth@ssyx.ucsc.edu (Steven Grimm)
Posting-number: Volume 1, Issue 68
Archive-name: uupc/part04

```
#!/bin/sh
# this is part 4 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file readme.por continued
#
CurArch=4
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> readme.por
X   uu and mail are also here.
X
X   This generic main program is used to start up both uu and
X   mail, by having the preprocessor symbol MAIN #defined to
X   different procedure names in the files which include host.c.
X   This generic main program should perform all the necessary
X   start up and wrap up functions as required by the host
X   operating system and call the routine MAIN while the host
X   environment is established.
X
X   If the preprocessor symbol CWDSPOOL is #defined by the file
X   that #includes host.c, the main program should change the
X   current working directory to the spooling directory before
X   calling MAIN, and switch back to the orginal directory after
X   MAIN had returned.
X
X   The following are the other routines residing in host.c and
X   used by the others parts of both uu and mail:
X
X   importpath - A deterministic function which maps a canonical
X      file name to a local file name.  This function must be
X      deterministic (i.e. always return the same local name when
X      given the same canonical name) because it is used on each
X      canonical file name several times in various part of the
X      code to obtain the corresponding local file name.
```

```
X
X      A canonical file name has the same format as a UNIX
X      pathname, and the format of a local file name is defined by
X      your local file system.
X
X      This function should at least preserve the first 6 and last
X      2 characters of the canonical file name, since this parts
X      of the canonical file name usually contain the site name
X      and the sequence number, which is critical to the
X      successful operation of uupc.
X
X      Perferably the last 4 characters of the canonical file name
X      should be perserved, since that is the number of digits in
X      the sequence number, but if that is not possible, an
X      attempt should be made to preverse the last 3 characters
X      before resorting to only preserve the last 2 characters.
X      (With only the last 2 characters preserved, the spool file
X      sequence number cycle will become only 100 before it goes
X      around again.)
X
X   mkfilename - Build a local path name out of a given local
X      directory path and local base name pair.  It usually just
X      concatenates the two parts together with the local system's
X      directory path separator between them.
X
X   loadenv - Retrieve certain configuration parameters from the
X      user's "environment" and make them available to the
X      program.  This involves filling in the appropriate global
X      (char *) variables to point to the appropriate strings
X      which contains the character value of the desired
X      configuration parameters.
X
X   If your system requires the differentiation of text file and
X   binary files, you should also supply the following routine.
X
X   filemode() - If this routine is passed the character 't' as
X      parameter, any subsequently opened file in the program
X      should be opened in text mode.  Similarly, if the parameter
X      is the character 'b', all subsequently opened file should
X      be opened in binary mode.
X
X
Xmlib.c - Library of routines used only by the mail part.
X
X   Currently there is only one routine in this library.
X
X   get_one - Wait for a single character to be typed on the
X      console keyboard and return to the caller with the
X      character read as soon as it is pressed.  It short, this is
X      a routine that detects and returns a keypress.
X
X      This routine is used when you reach the bottom of a page
X      while paging through your mail.
```

```
X
X      The single character read by this routine should not be
X      echoed to the console screen.
X
X
Xulib.c - Library of routines used only by the uu part.
X
X   login - The logger which verifies logins and passwords for
X      incoming UUCP connections.  You only need to have this
X      routine functional if you plan to ever run your node in
X      slave mode.  i.e. Waiting on the phone line for other nodes
X      to call you to make UUCP connections.
X
X   shell - Perform an UNIX command sent from a remote site via
X      uux.  To support incoming remote mail you need to emulate
X      the UNIX 'rmail' command, which can be easily done using
X      the pcmail package (compiled as rmail) which is part of
X      this uupc distribution.  If you want to also support
X      incoming USENET news-feeds, the UNIX 'rnews' command will
X      need to be supported as well.
X
X      If your system is capable of invoking another program
X      within a program, you might want to dispatch rmail and
X      rnews as a separate program here.  Otherwise, you might
X      need to compile, or link, them into uu as routines.
X
X   sleep - Wait a specified number of seconds in real-time.  You
X      could either use a busy wait loop or a timer alarm here,
X      depending on if your system has other (e.g. background)
X      jobs competing for CPU time at the same time.  On mutli-
X      tasking or multi-users systems, you would likely *not* want
X      to busy wait even if that's easier to implement.
X
X   The rest of this library consists of routines to deal with the
X   communications line (serial port).
X
X   openline - Open a communications line as the active line.  The
X      name of the serial line device and the speed to open the
X      line at are *both* specified as (char *) type parameters.
X      These value are just the corresponding values in the
X      systems file entry of the site being called.
X
X   sread - Read a specified number of bytes from the active
X      serial line and return with *no* input characters consumed
X      *if* the specified number of bytes is not available after
X      the specified timeout period.  This is basically a non-
X      blocking read that waits up to an user-specified amount of
X      time before returning.
X
X   swrite - Write a specified number of bytes out to the active
X      serial line.  No timeout mechanism needs to be provided by
X      this routine.
X
```

```
X   closeline - Close the active communications line.
X
X   SIOSpeed - Change the line speed of the active communications
X      line on-the-fly.  The new line speed is given as (char *)
X      rather than (int).  Note the mixed-case nature of this
X      routine name.
X
X
Xndir.h - Header file for the directory scanning routines.
X
X   At least the following needs to be defined in this file.
X
X   -- The constant 'MAXNAMLEN' declared with "#define".  This is
X      the maximum length of a file name in your system.  Note
X      that a file name does not include a directory path prefix,
X      it is only the maximum length of a file name within a
X      directory.
X
X   -- The structure 'direct' declared with "struct".  This
X      structure is a public data structure and its fields are
X      examined directly by uu, so your declaration needs to
X      provide at least the following fields:
X
X      struct direct {
X         short d_reclen;
X         short d_namlen;
X         char  d_name[MAXNAMLEN + 1];
X      };
X
X      d_reclen is the length of the structure minus the size of
X      the unused portion of d_name at the end of the structure.
X
X      d_namelen is strlen(d_name).
X
X      d_name is the name of the next file in the scan, with a
X      terminating '\0'.
X
X      It is important for the structure 'direct' to be defined
X      using "struct" insted of "typedef".
X
X   -- The type 'DIR' declared with "typedef".  This is a private
X      structure used by the ndir routines and should contain all
X      the static data related to a single invocation of the ndir
X      routines.
X
X      It is important for 'DIR' to be defined using "typedef"
X      instead of "struct".  (Note this is the *reverse* of the
X      structure 'direct' above.)
X
X   -- The routines opendir(), readdir(), and closedir().  Which
X      opens a specified directory, read the next file entry from
X      an opened directory, and close an opened directory,
X      respectively.
```

```
X
X
Xndir.c - "Berkeley-style" directory scanning routines.
X
X   This file contains a set of routines similar in functions to
X   the Berkeley-style directory scanning routines.  These
X   routines are used only by uu, not mail, and only the
X   opendir(), readdir(), and closedir() routines are used, and
X   therefore need to be implemented.
X
X   Eventhough the ndir routines should be capable of mutliple
X   concurrent invocations using separate (DIR *)'s, uu only uses
X   one invocation of it at a time.
X
X   If your system's file name is mono-case, then your readdir()
X   routine should always return the file name field (d_name) in
X   all lowercase.
X
X
XAny questions about porting uupc to other machines/operating
Xsystems, and comments and suggestions about this writeup should
Xbe directed to one of the e-mail addresses listed at the end of
Xthis file.
X
XIf you do decide to start porting uupc to a new machine/operating
Xsystem, please drop us a line as well.  We might even be able to
Xsave each other some duplicated efforts!
X
X
XUUCP: {seismo,ihnp4!alberta,uw-beaver,uunet}!ubc-vision!van-bc!uupc
XInternet: uupc@van-bc.UUCP
X-------
X
X
X--
X{ihnp4!alberta!ubc-vision,uunet}!van-bc!Stuart.Lynne Vancouver,BC,604-937-7532
X
SHAR_EOF
chmod 0600 readme.por || echo "restore of readme.por fails"
sed 's/^X//' << 'SHAR_EOF' > readme.st &&
XThis file contains the Atari ST specific files.
X
XI have done all the compilation using the Alcyon C versin 4.14a compiler.
XThe disks were layed out with all the sources and objects on the A drive and
Xthe compiler on the D drive.
X
XProblems with the Atari ST code:
X
X    - there is a problem in the getsystem(), sysend() states such that
X      if there is more than one system in the systems file then the
X      program may bomb bomb (bus error?)
X
XFiles included in this shar script:
```

```
X
X        On the atari these files should be placed in the
X        \local directory or the .h files will need to be
X        edited.
X
X      dirsubs.c   - file and directory name manipulation
X      dmymath.c   - dummy floating point routines to fake out linker
X      genv.c  - load enviroment strings
X      getenv.c    - get and enviroment string
X      getone.c    - get one character without echo
X      host.c  - host specific main program
X      inoutpth.c  - file name conversion
X      ndir.c  - directory manipulation routines
X      printmsg.c  - print message to log file
X      time.c  - date/time conversion routines
X      uusup.c - uu support routines
X
X      The following files belong in the ./local dirctory and should
X      be moved there with the leading z stripped off.
X
X      zdcp.h  - main include file for uu
X          zmailhost.c  - host specific mail main program
X          zuuhost.c    - host specific uu main program
X      zgenv.h - definitions for enviroment variables
X      zhost.h - host specific definitions
X      zndir.h - directory manipulation routines header
X      ztime.h - time routine header
X
XThe following script files are included as examples on how to compile
Xthe sources.  They are for the CM command shell.
X
X      c.mm        - compile script for alcyon C (file is in A:\)
X      cu.mm       - compile script for alcyon C (file is in A:\atari\)
X      env.mm  - define enviroment script
X      ldcp.mm - link uu.prg
X      lmail.mm    - link mail.prg
X      uuar.mm - create DCPLIB
X      utar.mm - create UTLLIB
X
XBasic procedure is to:
X    - compile all the .c files into .o files
X    - build the DCPLIB with uuar.mm
X    - build the UTLLIB with utar.mm
X    - link uu with ldcp.mm
X    - link mail with lmail.mm
X
XIf you have problems feel free to drop me a line.  Though I warn you I have
Xa slow turn around time on questions I will try to get back to you within
Xa day or two.
X
X-------------------------------------------------------------------------
XUUCP:  {alberta,uw-beaver,siesmo}!ubc-vision!van-bc!nvanbc!lawrence
XSNAIL: 733 Sylvan Ave., North Vancouver, B.C., Canada, V7R 2E8
```

```
XPHONE: 1-604-736-9241 (09:00-17:00 PDT)
X
SHAR_EOF
chmod 0600 readme.st || echo "restore of readme.st fails"
sed 's/^X//' << 'SHAR_EOF' > readme.uup &&
XJuly 29/87                          Stuart Lynne
X
XUUPC is now running on Macintosh, Atari-ST, Amiga, and IBM-PC with MS-DOS.
X
XThese versions will constitute my first release of uupc and pcmail. Please
Xsee the README files in each shar file for appropriate instructions.
X
XAs is, uupc sends and receives files quite well. Still unimplemented is
Xthe reverse direction file transfers, i.e. send a file to a remote host
Xwhile in slave mode, receive a file while in master mode. These are not
Xneeded to support news and mail. A proper uucp command is need too.
X
XThe user agent mail program is a very simple hack to simply allow you to
Xread and send mail. Hopefully someone will work on replacing this!
X
XThe message transfer agent pcmail program is fairly robust. It does need to
Xhave some more intelligence to allow for more intelligent routing of outgoing
Xmail. Currently ALL outgoing mail is forward to a single remote site for
Xprocessing. This will actually handle the needs of a large number of users
Xbut for those lucky ones who can actually get access to several large uucp
Xsites better routine would be nice.
X
X
XOther things which are needed:
X
X    news unbatcher
X    news reader
X
XCurrently the news is simply spooled to a directory with a unique file name.
XYou can read this with a normal text editor. If you wish to have outgoing mail
Xthe easiest way is to have your news feed set up alias's like:
X
X    comp.sys.amiga "| /usr/local/lib/news/recnews comp.sys.amiga"
X
XThen simply mail your article to:
X
X    comp.sys.amiga@newsfeedhost.uucp
X
X
XPlease feel free to port this code to other environments. I ask only that you
Xtry and limit changes to the machine independent code. Things have been setup
Xso that you shouldn't have to modify dcp, uupc, mail or pcmail if you are simply
Xporting to a new environment.
X
XPlease send me any new versions, diff's to get old versions working better,
Xbug fixes etc.
X
XHave fun and enjoy.
```

```
X
XI would suggest that questions and comments about uupc/dcp/mail be directed
Xto comp.mail.uucp on Usenet. This group is about "Mail in the uucp network
Xenvironment." which describes uupc pretty well.
X
X
XQuestions, bug fixes etc
X
Xuupc, mac version, information
X
X    Stuart Lynne
X    stuart.lynne@van-bc.uucp
X    {{seismo,ihnp4!alberta}!ubc-vision,uunet}!van-bc!Stuart.Lynne  604-937-7532
X
XAmiga version
X
X    Jeff Lydiatt
X    jl@jlami.vnet.van-bc.uucp
X    {{seismo,ihnp4!alberta}!ubc-vision,uunet}!van-bc!jlami!jl
X
XAtari ST version
X
X    Lawrence Harris
X    lawrence@nvanbc.uucp
X    {{seismo,ihnp4!alberta}!ubc-vision,uunet}!van-bc!nvanbc!lawrence
X
XIBM PC - MS-DOS version
X
X    Samuel Lam
X    skl@sklpc.vnet.van-bc.uucp
X    {{seismo,ihnp4!alberta}!ubc-vision,uunet}!van-bc!skl
X
Xuupc mailing list
X
X    uupc@van-bc.uucp            Automatically forwarded to mailing list
X    uupc-request@van-bc.uucp    For requests to be added to mailing list
X
X    {{seismo,ihnp4!alberta}!ubc-vision,uunet}!van-bc!nvanbc!uupc
X    {{seismo,ihnp4!alberta}!ubc-vision,uunet}!van-bc!nvanbc!uupc-request
X
X
XNB. Rurrently UUNET is only polled twice per week, you may wish to send
Xany messages via both paths to ensure delivery. UUNET is probably more
Xreliable, ubc-vision may be faster if you can reach them from your site.
X
X
X
Xuupc              June 8, 1987       Stuart Lynne
X
XFor Beta implementors only.
X
Xuupc incorporates a streamlined version of dcp to implement a uucp mail
Xand news delivery system.
```

```
X
XSee README.DCP for dcp info.
X
XBy moving the host dependant code into one file the other four dcp files can
Xhopefully be maintained easily. It should be possible to maintain one version
Xof them which compiles and runs on all machines without conditional compilation
Xflags.
X
XThe host file should contain:
X
X    - serial I/O
X    - BSD compatible directory routines
X    - system call stuff
X
XThis all goes to implement a command called uupc. It is similiar to the uucico
Xcommand under unix.
X
X    uupc [-xn] [-shost]
X
X
XThere is also two mail source files. Pcmail is the MTA part of the mail system.
XIt compiles in two ways, one for rmail (add only From and Received: headers),
Xdefine a simple rnews; and for lmail (add all headers). Pcmail will effect
Xdelivery of mail to local and remote users. Currently all remote mail is
Xdirected to one smart host for forwarding.
X
XMail is a simple UA. It allows you to send mail and read your mailbox. It needs
Xlots of work but is servicable.
X
X    mail -s "subject here" user user@remote.site.domain < message
X
X    mail -f =mailbox
X
X    mail
X
Xwill all do the obvious. Mail will append a .signature if it can find one, and
Xwill keep a copy of your outgoing mail in =mail.sent.
X
XToDo:
X
X    uucp command
X    mail improvements
X    bug fixes to uupc/dcp
X    ports to Atari, Amiga, IBM PC, VMS
X
X
XMakefile - sample Makefile (Macintosh Aztec C)
Xgetargs.c    - library routine
Xlmail.c  - define LMAIL; include pcmail
Xmail.c   - mail program (UA)
Xpcmail.c     - mail program (MTA)
Xrmail.c  - define RMAIL; include pcmail
Xlib.c        - misc library routines, FOPEN, CREAT, getargs, printmsg
```

```
X
Xhost.h       - prototype for host.c, includes "local/host.h"
Xmailhost.c   - ditto for mailhost.c, includes "local/host.c"
Xmlib.c       - ditto for mlib.c,    includes "local/mlib.c"
Xulib.c       - ditto for ulib.c,    includes "local/ulib.c"
Xuuhost.c     - ditto for uuhost.c,  includes "local/host.c"
X
X
Xpcmail in general
XPcmail provides MTA functionality. It delivers the mail. Currently it is
Xvery dumb about forwarding mail. Local deliveries always succeed if there
Xis room and the mail spooling directory exists. No "/etc/passwd" file list
Xof users is used to determine if there really is a mailbox for an incoming
Xmessage. Also outgoing mail is assumed to go to one smart host for
Xprocessing. This is determined by scanning for "!" or "@" in the address.
X
XBoth local and remote delivery algorithms could be souped up. Locally we
Xshould maintain a list of mailboxes. For remote we should attempt to build
Xa path to the most reasonable host for forwarding a specific message. This
Xwill require a small version of the paths database most likely. If your
Xonly talking to one host anyway the current scheme is not all that bad.
X
XPcmail has one additional capability which is not currently being exploited.
XIt can add additional message length header lines and spool mail into a
Xmailbag. This mailbag could then be sent intact to your host for processing.
XThe host must run rpcmail (availble from sl@van-bc.uucp) to unbatch the
Xmessages to rmail. This corresponds to the AT&T Mail protocol for uploading
Xmail from PC's.
X
XMost likely the converse capability would be more suitable. Have the host
Xbatch incoming mail and news. Unbatch it on this side.
X
Xpcmail / LMAiL
XThe mail UA is composed of the mail.c program and pcmail.c compiled without
Xdefining RMAIL (LMAIL). The LMAIL version of pcmail adds rfc822 headers
Xto all mail, copies mail to =mail.sent etc.
X
Xpcmail / RMAIL
XThe uu program contains the RMAIL version of pcmail. It only adds the From
XReceived: headers to incoming mail.
X
X
X
XFor more information, bug fixes, more commands etc:
X
X        Stuart.Lynne@van-bc.uucp
X        604-937-7532
X
X
XDirectory tree
X
X/usr
X/usr/lib
```

```
X/usr/lib/uucp
X/usr/lib/uucp/SEQF             - sequence numbers
X/usr/lib/uucp/systems      - host connection information
X/usr/spool
X/usr/spool/mail                - mail directory
X/usr/spool/mail/XXXX       - user mail files
X/usr/spool/rnews           - rnews spool/file
X/usr/spool/uucp                - spool directory
X/usr/spool/uucp/C.YYYYYNNNN    - copy control files
X/usr/spool/uucp/D.YYYYYNNNN    - uucp data files
X/usr/spool/uucp/dcp.log        - log file
X/usr/spool/uucp/X.YYYYYNNNN    - execute control files
X/usr/XXXX                      - user directories
X/usr/XXXX/.signature       - signature file
X/usr/XXXX/Mail                 - user mail directory
X/usr/XXXX/Mail/mail.sent   - sent file
X/tmp                           - temporary files
X
X
X
XSystems File
X
XNB. I have split the lines, in the real file they should be one line for
Xeach entry.
X
XThis entry uses the built in Hayes dialer.
X
Xvan-bc Any a HAYES TD939-4782 g ogin:--ogin: uuslmac sword:-\c-sword: uuslmac
X
X    Connect to van-bc using HAYES dialer with phone number 939-4782 with
X    protocol g.
X    Wait for ogin: if timeout send \n and wait for ogin:.
X    Send uuslmac (login ID).
X    Wait for sword:, if timeout send nothing, wait for sword:
X    Send uuslamc (login ID).
X
XI use this entry to connect via a DC Hayes Smartmodem 2400, dialing explicity.
X
Xvan-bc Any a DIR 2400 g "" ATZ OK-\d+++\dATZ-OK ATS7=12 OK ATTD939-4782
X    CONNECT \d\c ogin:--ogin: uuslmac sword:-\c-sword: uuslmac
X
X    Connect to van-bc at 2400 on port a with protocol g.
X    Expect nothing, send ATZ to reset the modem,
X    Expect OK, if not received send pause +++ pause to try and get
X    modems attention and wait for OK.
X    When OK received, send ATS7=12 to shorten connect timeout on modem.
X    Expect OK, send ATTD939-74782.
X    Expect CONNECT, send nothing but pause (\c is needed).
X    Expect ogin:, if not received send newline.
X    Send login name.
X    Expect sword:, send password.
X
X
```

```
XThis entry is used to connect directly to my Callan at 9600. It is
Xcomplicated due to the Callan running a special mgetty program which
Xthinks it is talking to a Hayes Smartmodem. So we fake it out, then
Xtell it the connection has been made at 9600, then switch to 9600
Xourselves.
X
Xvan-bc Any a DIR 2400 g "" OK\r\d\dRING\r\dCONNECT\s9600\d\z9600\
X    ogin:-\r\r-ogin: uuslmac sword:-\c-sword: uuslmac "" \d\d\d\d\d\d\c
X
X    Connect to van-bc at 2400 on port a.
X    Expect nothing, send OK, pause, RING, CONNECT, 9600,
X    and change to 9600 bps.
X    Expect ogin: send login id.
X    Expect sword: send password.
X
X
X
SHAR_EOF
chmod 0600 readme.uup || echo "restore of readme.uup fails"
sed 's/^X//' << 'SHAR_EOF' > rmail.c &&
X/*           rmail.c
X
X*/
X
X#define NOMAIN
X#define RMAIL /* if defined we get rmail() else we get lmail() */
X
X#include "pcmail.c"
X
X
SHAR_EOF
chmod 0600 rmail.c || echo "restore of rmail.c fails"
sed 's/^X//' << 'SHAR_EOF' > shell.c &&
X
X
SHAR_EOF
chmod 0600 shell.c || echo "restore of shell.c fails"
sed 's/^X//' << 'SHAR_EOF' > systems &&
Xssyx Evening a HAYES DT1112222 g ogin:--ogin:--ogin:--ogin:--ogin: uholo sword:-
\c-sword: [password]
Xdaver Evening a HAYES DT9876543 g "" \d\r ogin:-\r-ogin: Uholodek ord: [password]
Xsmegma Any a HAYES DT3141592 g "" \r\r\r\r ogin:~1-\r-ogin:~1-\r-ogin:~1-\r-ogin:
Uholodek ord: [password]
SHAR_EOF
chmod 0600 systems || echo "restore of systems fails"
sed 's/^X//' << 'SHAR_EOF' > time.c &&
X/*
X * time(), ctime(), localtime(), sleep()
X *
X */
X#include <osbind.h>
X#include "time.h"
X#define NULL 0
```

```
X
Xlong *_hz200 = (long *)0x0004ba;     /* address of system 200 hz clk
     */
Xlong _prtime;
X
X/*
X * read200hz() - read the system 200hz clock
X *
X * Must be a separate routine since it is called in supervisor mode
X */
Xvoid _read200hz()
X{
X    _prtime = *_hz200;
X}
X
Xlong time( tloc )
Xlong *tloc;
X{
X    long t;
X    Supexec( _read200hz );
X    t = _prtime / 200L;
X    if ( tloc != (long *)NULL ) *tloc = t;
X    return( t );
X}
X/*
X * ctime(clock) - Return the time as DDD MMM dd hh:mm:ss YYYY\n\0
X *
X * Since I don't have a working time() call just now I ignore the clock
X * and simply return the current date and time.  This is all that this
X * program requires anyway.
X *
X * This routine was rewritten by Steven Grimm, since it didn't work too
X * well as supplied (in fact, it didn't work at all!)
X */
Xchar _timearr[26];
Xchar *_Months[12] = {
X    "Jan", "Feb", "Mar", "Apr", "May", "Jun",
X    "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"
X};
X/* Days of the week.  Start with Tuesday so we eliminate one addition. */
Xchar *_Days[7] = { "Tue", "Wed", "Thu", "Fri", "Sat", "Sun", "Mon" };
X
X/* Offset into the year for 1st of each month (disregarding leap years) */
Xint monthoff[12]={ 0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334 };
X
Xtypedef struct {
X    unsigned day:5;
X    unsigned month:4;
X    unsigned year:7;
X} GDATE;
Xtypedef struct {
X    unsigned second:5;
X    unsigned minute:6;
```

```
X    unsigned hour:5;
X} GTIME;
X
Xchar *ctime( clock )
Xlong *clock;
X{
X    unsigned date;
X    unsigned time;
X    long dayno;
X    GDATE gdate;
X    GTIME gtime;
X
X    date = Tgetdate() & 0xffffL;     /* Get date from DOS */
X    time = Tgettime() & 0xffffL;     /* Get time from DOS */
X    gdate = *(GDATE *)&date;
X    gtime = *(GTIME *)&time;
X
X/*
X * Now we have to figure out the day of the week.  We do this by starting
X * at a known date -- in this case Tuesday, January 1, 1980 -- and counting
X * days from there.  This formula is optimized a bit, so may look somewhat
X * cockeyed.
X */
X    dayno = gdate.year * 365 +  /* 365 days per year since '80 */
X         gdate.year / 4 +   /* add leap days */
X         monthoff[gdate.month-1]+ /* and offset to this month */
X         gdate.day;         /* and the day */
X    if (((gdate.year % 4)==0) && (gdate.month < 3))
X         dayno--;
X    dayno %= 7;
X
X    sprintf( _timearr, "%s %s %02d %02d:%02d:%02d PDT %04d\n",
X              _Days[dayno],
X              _Months[gdate.month - 1],
X              gdate.day, gtime.hour, gtime.minute,
X              gtime.second * 2, gdate.year + 1980);
X    return( _timearr );
X}
X
X/*
X * localtime() - Return tm structure (for current time/date, oh well).
X */
X
Xstruct tm _TMstruct;
X
Xstruct tm *localtime( clock )
Xlong *clock;
X{
X    unsigned date;
X    unsigned time;
X    long dayno;
X    GDATE gdate;
X    GTIME gtime;
```

```
X
X    date = Tgetdate() & 0xffffL;     /* Get date from DOS */
X    time = Tgettime() & 0xffffL;     /* Get time from DOS */
X    gdate = *(GDATE *)&date;
X    gtime = *(GTIME *)&time;
X
X    dayno = gdate.year * 365 +  /* 365 days per year since '80 */
X         gdate.year / 4 +   /* add leap days */
X         monthoff[gdate.month-1]+ /* and offset to this month */
X         gdate.day;         /* and the day */
X    if (((gdate.year % 4)==0) && (gdate.month < 3))
X         dayno--;
X    dayno = (dayno + 5) % 7;
X
X    _TMstruct.tm_sec = gtime.second * 2;
X    _TMstruct.tm_min = gtime.minute;
X    _TMstruct.tm_hour= gtime.hour;
X    _TMstruct.tm_mday= gdate.day;
X    _TMstruct.tm_mon = gdate.month;
X    _TMstruct.tm_year= gdate.year + 80;
X    _TMstruct.tm_wday= dayno;
X    _TMstruct.tm_yday= monthoff[gdate.month-1]+gdate.day;
X    _TMstruct.tm_isdst= 0;
X    return( &_TMstruct );
X}
X
X/*
X * sleep() - wait n seconds
X *
X * Simply delay until n seconds have passed
X *
X */
X
Xvoid sleep(n)
Xunsigned int n;
X{
X    unsigned long i;
X
X    i = (unsigned long)time((long *)NULL);
X    while ( ( (unsigned long)time((long *)NULL) - i ) < n ) ;
X}
SHAR_EOF
chmod 0600 time.c || echo "restore of time.c fails"
sed 's/^X//' << 'SHAR_EOF' > time.h &&
X/* time.h */
X
Xstruct tm {
X    int  tm_sec;       /* time of day, seconds */
X    int  tm_min;       /* time of day, minutes */
X    int  tm_hour;  /* time of day, hours (24 hour clock) */
X    int  tm_mday;  /* day of month (1-31) */
X    int  tm_mon;       /* month of year (0-11) */
X    int  tm_year;  /* year - 1900 */
```

```
X    int  tm_wday;  /* day of week (Sunday = 0) */
X    int  tm_yday;  /* day of year (0-365) */
X    int  tm_isdst; /* non-0 if DST in effect */
X};
Xextern struct tm *localtime();
Xextern char *ctime();
X
SHAR_EOF
chmod 0600 time.h || echo "restore of time.h fails"
sed 's/^X//' << 'SHAR_EOF' > ulib.c &&
X/*      ulib.c
X
X*/
X
X#include "local/ulib.c"
X
SHAR_EOF
chmod 0600 ulib.c || echo "restore of ulib.c fails"
sed 's/^X//' << 'SHAR_EOF' > uuhost.c &&
X/*      uuhost.c
X
X*/
X#define  MAIN dcpmain
X#define CWDSPOOL
X#include "host.c"
X
X
SHAR_EOF
chmod 0600 uuhost.c || echo "restore of uuhost.c fails"
sed 's/^X//' << 'SHAR_EOF' > uusup.c &&
X/*
X * atari st system specific subroutines for uu
X *
X * these are included into uuhost.c during compile
X */
X#include "dcp.h"
X
Xstruct
X   {
X      char   *baud;
X      int    baudidx;
X   } BaudStr[] = { "300", B300,
X           "1200", B1200,
X           "2400", B2400,
X           "4800", B4800,
X           "9600", B9600,
X           "19200", B19200,
X           "-1", -1
X         };
X
X/* some global variables */
X
Xchar st_ibuf[IBUFSIZ];          /* our own input buffer
```

```
     */
Xchar st_obuf[OBUFSIZ];          /* and our own output buffer
     */
X
X/* most of the Iorec() stuff was redone by Martin Minow (thanks) */
X
XIOREC *st_sysr;                 /* ptr to system rs232 record
     */
XIOREC st_savr;                  /* to save system rs232 record
     */
XIOREC st_myiorec;
X
Xunsigned long     old_rsconf;
Xint      old_ucr, old_rsr, old_tsr, old_scr;
Xint      st_baud = B9600;   /* default baud rate */
X
X/*
X * r_count_remaining() - Access the i/o structures and find out how many
X * characters there are in the input ring buffer.
X */
X
Xint r_count_remaining()
X{
X    IOREC *ReadPort;
X    int Diff;
X
X    ReadPort = (IOREC *)Iorec(0);
X    Diff = ReadPort->in.ibuftl - ReadPort->in.ibufhd;
X    if ( Diff < 0 )
X      Diff = IBUFSIZ - abs( Diff );
X    return( Diff );
X}
X
X/*
X *                       Read/Write I/O processing
X */
X
X/*
X * openline - Open the communication line for use with dcp.
X */
X
Xopenline( name, baud )
X    char *name, *baud;
X{
X
X     register int i;
X
X
X     st_myiorec.in.ibuf = st_ibuf;
X     st_myiorec.in.ibufsize = IBUFSIZ;
X     st_myiorec.in.ibufhd = st_myiorec.in.ibuftl = 0;
X     st_myiorec.in.ibuflow = IBUFSIZ / 4;
X     st_myiorec.in.ibuflow = (IBUFSIZ / 4 ) * 3;
```

```
X
X     st_myiorec.out.ibuf = st_ibuf;
X     st_myiorec.out.ibufsize = OBUFSIZ;
X     st_myiorec.out.ibufhd = st_myiorec.out.ibuftl = 0;
X     st_myiorec.out.ibuflow = OBUFSIZ / 4;
X     st_myiorec.out.ibuflow = (OBUFSIZ / 4 ) * 3;
X
X     for (i=0;    BaudStr[i].baudidx != -1; i++ )
X       if ( strcmp( baud, BaudStr[i].baud) == 0 )
X     {
X       st_baud = BaudStr[i].baudidx;
X       break;
X      }
X/*
X *  set up our own rs232 input and output buffers
X */
X    st_sysr = (IOREC *)Iorec(0);
X    st_savr = *st_sysr;             /* Save system buffer
     */
X    *st_sysr = st_myiorec;      /* Set my io buffer    */
X
X/*
X * Saving the old Rsconf() parameters, and
X * adding the 2 Vsync() calls was done by Martin Minow.
X *
X * Rsconf(baud, ctrl, usr, rsr, tsr, scr):
X * baud  New line speed
X * ctrl  UART control register: 2 stop bits, match speed
X * rsr   Receiver on
X * tsr   Transmitter on
X * scr   Synchronous register ignored.
X * Rsconf returns the "old" values.
X */
X
X    printf("Setting Com port - speed = %d\n", st_baud);
X    old_rsconf = Rsconf(st_baud, NOFLOW, UCR, RSR, TSR, SCR);
X    old_ucr = (old_rsconf >> RS_UCR_SHIFT) & 0xFF;
X    old_rsr = (old_rsconf >> RS_RSR_SHIFT) & 0xFF;
X    old_tsr = (old_rsconf >> RS_TSR_SHIFT) & 0xFF;
X    old_scr = (old_rsconf >> RS_SCR_SHIFT) & 0xFF;
X/*
X * Magic.
X */
X    Vsync();          /* Short delay while   */
X    Vsync();          /* The UART settles    */
X    return ( 0 );
X}
X
X/*
X * closeline - Restore the communication line to the way we found it.
X */
X
Xvoid closeline()
```

```
X{
X/*
X * Drop DTR for a second so we hang up.
X */
X    Ongibit(0x10);
X    sleep(1);
X    Offgibit(0xffef);
X    /* restore the system's i/o record                 */
X    *st_sysr = st_savr;
X
X    /* Replace old RS232 parameters
X    Rsconf(B9600, 1, old_ucr, old_rsr, old_tsr, old_scr);*/
X}
X
X/*
X * sread( buf, num, timeout ) - Read num characters within timeout
X */
X
Xint sread( buf, num, timeout )
Xunsigned char *buf;
Xint  num, timeout;
X{
X    int count,test;
X    unsigned char *cp;
X
X    if (debuglevel > 13)
X      fputc( '[', stderr );
X    printmsg(15, "sread: num: %d  timeout: %d", num, timeout );
X
X    count = xread( buf, num, timeout );
X
X    printmsg(15, "sread: read: %d ", count);
X
X    if (debuglevel > 13 && count > 0)
X    {
X      test = count;
X      cp = buf;
X      while (test--)
X        fprintf( stderr, isprint(*cp)? "[%c]":"[%02x]", *cp++ );
X    }
X    if (debuglevel > 13)
X      fputc( ']', stderr );
X    return( count );
X}
X
X/*
X * swrite( buf, num ) - Write num characters to the output port
X */
X
Xint swrite( buf, num )
Xint  num;
Xchar *buf;
X{
```

```
X    int  count;
X    char *cp;
X
X    if (debuglevel > 14)
X    {
X      fputc( '{', stderr );
X      count = num;
X      cp = buf;
X      while ( count-- )
X        fprintf( stderr, isprint(*cp)? "{%c}":"{%02x}", *cp++ );
X      fputc( '}', stderr );
X    }
X    count = xwrite( buf, num );
X    return( count );
X}
X
X/*
X * read buffer - low level
X */
X
Xint xread( buf, num, timeout )
Xchar buf[];
Xint  num, timeout;
X{
X    long jtime;
X    int i,jj,jd;
X
X    jtime = time((long *)NULL);
X    while (TRUE)
X    {
X      jj = r_count_remaining();
X      if ( jj >= num )
X      {
X        for ( i=0; i < num; i++ )
X        {
X          buf[i] = Bconin(AUX) & 0xFF;
X        }
X      return( jj );
X      }
X      else
X      {
X        jd = time((long *)NULL) - jtime;
X        if ( jd >= timeout )
X          return( jj );
X      }
X    }
X}
X
X/*
X * xwrite( buf, n ) - Write n bytes froum the buffer to the port
X */
X
Xint xwrite( buf, n )
```

```
Xchar buf[];
Xint  n;
X{
X    register int i;
X
X    for ( i=0; i<n; i++ )
X    {
X      Bconout( AUX, buf[i] );
X    }
X    return( n );
X}
X
X/*
X * Set I/O port to a certain baud rate.
X * Steven Grimm, 7-88
X */
Xvoid SIOSpeed( s )
Xchar *s;
X{
X    int i;
X
X    for (i=0; BaudStr[i].baudidx != -1; i++ )
X         if ( strcmp( s, BaudStr[i].baud) == 0 )
X         {
X              Rsconf(BaudStr[i].baudidx, -1, -1, -1, -1, -1);
X              break;
X         }
X    return;
X}
X/*
X *                  Command shell processing
X *
X * Pass a command the the host operating system to execute.
X *
X */
X/*      shell
X
X
X*/
Xnotimp( argc, argv )
Xint argc;
Xchar **argv;
X{
X    printf( "%s not implemented\n", argv[0]);
X}
X
Xchar *getcwd();
X
Xshell( command, inname, outname, errname )
Xchar * command;
Xchar * inname;
Xchar * outname;
Xchar * errname;
```

```
X{
X
X    char *argvec[50];  /* array of pointers to make up arg list */
X    char hostname[32]; /* place to put munged file name */
X    char curdir[64];   /* place to remember current directory */
X
X    int  rmail(); /* mail program */
X    int  rnews(); /* news program */
X    int  (*proto)();   /* variable containing program pointer */
X
X    int  argc;         /* will contain the number of args */
X    char **argv;       /* will contain pointer to arg vector */
X
X    int  args;         /* temporary for copy of argc */
X
X    /*
X     * Break up the command line into an argument vector
X     */
X    argc = getargs( command, argvec );
X    argv = argvec;
X
X    if ( debuglevel > 5 )
X    {
X       args = argc;
X       while ( args )
X          fprintf( stderr, "arg: %d  %s\n", args--, *argv++ );
X       argv = argvec;
X    }
X    /*
X     * Initialize to function which will print 'not implemented
X     * then see if the command is one we know about and if so set
X     * up out function pointer.
X     */
X    proto = notimp;
X
X    if ( strcmp( *argv, "rmail" ) == SAME )
X       proto = rmail;
X    else if ( strcmp( *argv, "rnews" ) == SAME ) {
X       proto = rnews;
X    }
X
X    *hostname = '\0';
X    if ( *inname != '\0' )
X       importpath( hostname, inname );
X
X    if ( *hostname != '\0' ) {
X       fprintf( stderr, "reopening stdin as %s\n", hostname);
X       getcwd(curdir, 0);
X       fclose( stdin );
X       if ( fopen( hostname, "r" ) == (FILE *)NULL ) /* */
X/*     if ( freopen( hostname, "r", stdin ) == (FILE *)NULL ) /* */
X       {
X          fprintf( stderr, "Couldn't open %s, %d\n", hostname, errno );
```

```
X          exit(-1);
X       }
X    }
X    (*proto)( argc, argv );
X
X    chdir( curdir );
X
X    fclose( stdin );
X    if ( fopen(  "CON:", "r" ) == (FILE *)NULL )
X    {
X       fprintf( stderr, "Couldn't reopen CON:" );
X    }
X/*  freopen( "CON:", "r", stdin ); /* */
X
X}
X
X
X/* */
X/*
X *
X *      login (for slave in PC mode)
X * Real dumb login handshake
X*/
Xlogin()
X{
X    char logmsg[132];
Xlretry:
X    msgtime = 9999;
X    rmsg(logmsg, 0); /* wait for a <CR> or <NL> */
X    msgtime = 2 * MSGTIME;
X    wmsg("Username:", 0);
X    rmsg(logmsg, 0);
X    printmsg( 0, "Username = %s", logmsg );
X    wmsg("Password:", 0);
X    rmsg(logmsg, 0);
X    printmsg( 14, "Password = %s", logmsg );
X    if (strcmp(logmsg, "uucp") != 0)
X         goto lretry;
X    return('I');
X}
X
X
SHAR_EOF
chmod 0600 uusup.c || echo "restore of uusup.c fails"
rm -f s2_seq_.tmp
echo "You have unpacked the last part"
exit 0


From 3664470960921008775
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,dd73be6c8fa1c8d5,start
X-Google-Attributes: gid100493,public
```

X-Google-ArrivalTime: 1988-08-11 21:15:48 PST
Path: utzoo!utgpu!attcan!uunet!husc6!think!bloom-beacon!gatech!rutgers!ucsd!
ucbvax!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i069: gnufp -- Floating-point routines
Keywords: 68000, gcc
Message-ID: <4370@saturn.ucsc.edu>
Date: 2 Aug 88 16:24:28 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 773
Approved: koreth@ssyx.ucsc.edu

Submitted-by: jsm@phoenix.princeton.edu (Scott McCauley)
Posting-number: Volume 1, Issue 69
Archive-name: gnufp

[This will probably require some tweaking before it will compile on the
 ST. At best, you'll have to adapt gcc's inline assembly to whatever
 your compiler uses. -sg]

Due to popular demand, here is gnulib.c, a set of floating point routines
for the 68000 and 68020.

```
/* Subroutines needed by GCC output code for the 68000/20.

   Compile using -O flag with gcc.
   Use the -m68000 flag if you have a 68000

   This package includes 32 bit integer and 64-bit floating
   point routines.

   WARNING: alpha-test version (July 1988) -- probably full of bugs.
   If you find a bug, send bugs reports to jsm@phoenix.princeton.edu,
   or

        Scott McCauley
        PPPL P. O. Box 451
        Princeton NJ 08543

   Known bugs/features:

   1) Depending on the version of GCC, this may produce a lot of
      warnings about conversion clashes, etc. It should still compile
      as is. Also, there appears to be a bug in the register-allocation
      parts of gcc that makes the compiler sometimes core dump
      or run into an infinite loop. This version works -- if you
      decide to change routines these compiler bugs can bite very hard....

   2) all single-precision gets converted to double precision in the
      math routines (in accordance with K&R), so doing things in
      double precision is faster than single precision....
```

```
   3) double precision floating point division may not be accurate to
      the last four or so bits. Most other routines round to the
      lsb.

   4) no support of NaN and Inf

   5) beware of undefined operations: i.e. a float to integer conversion
      when the float is larger than MAXIINT.

*/

#include <stdio.h>
#include <math.h>


#include "config.h"

/* These definitions work for machines where an SF value is
   returned in the same register as an int.  */

#ifndef SFVALUE
#define SFVALUE int
#endif

#ifndef INTIFY
#define INTIFY(FLOATVAL)  (intify.f = (FLOATVAL), intify.i)
#endif

union double_di { double d; int i[2]; };
union flt_or_int { int i; float f; };

#ifdef WORDS_BIG_ENDIAN
#define HIGH 0
#define LOW 1
#else
#define HIGH 1
#define LOW 0
#endif

/* start of symbolic asm definitions */

/* you may have to change the g's to d's if you start getting
   illegal operands from as */

#define MUL(a, b) asm("mulu %2,%0" : "=d" (b) : "0" (b) , "g" (a))
#define DIV(a, b) asm("divu %2,%0" : "=d" (b) : "0" (b) , "g" (a))
#define SWAP(a) asm("swap %0" : "=r" (a) : "r" (a) , "r" (a) )

#define ASL(r1, r2) { asm("asll #1,%0" : "=d" (r2) : "d" (r2));\
                asm("roxll #1,%0" : "=d" (r1) : "d" (r1)); }
#define ASR(r1, r2) { asm("asrl #1,%0" : "=d" (r1) : "d" (r1));\
                asm("roxrl #1,%0" : "=d" (r2) : "d" (r2)); }
#define ADD(r1, r2, r3, r4) \
```

```
     { asm("addl %2,%0": "=g" (r4) : "0" (r4) , "g" (r2)); \
       asm("addxl %2,%0": "=g" (r3) : "0" (r3) , "g" (r1)); }

#define NEG(r1, r2) { asm("negl %0" : "=d" (r2) : "d" (r2)); \
                asm("negxl %0" : "=d" (r1) : "d" (r1)); }

#define umultl _umulsi
#define multl _mulsi3
#define udivl _udivsi3
#define divl _divsi3
#define ddiv _divdf3
#define qmult _qmult
#define dmult _muldf3
#define dneg _negdf2
#define dadd _adddf3
#define dcmp _cmpdf2
#define dtoul _fixunsdfsi
#define dtol _fixdfsi
#define ltod _floatsidf

/* quasi-IEEE floating point number definitions */

struct bitfloat {
     unsigned sign : 1;
     unsigned exp : 8;
     unsigned mant : 23;
} abfp;

struct bitdouble {
     unsigned sign : 1;
     unsigned exp : 11;
     unsigned mant1 : 20;
     unsigned long mant2;
};

#ifdef L_eprintf
#include <stdio.h>
/* This is used by the `assert' macro.  */
void
_eprintf (string, line)
     char *string;
     int line;
{
  fprintf (stderr, string, line);
}
#endif

#ifdef L_umulsi3

/*_umulsi3 (a, b) unsigned a, b; { return a * b; } */

unsigned long umultl(a,b)
unsigned long a, b;
```

```
{
     register unsigned long d7, d6, d5, d4;

     d7 = a;
     d6 = b;
     d5 = d6;
     d4 = d6;
     /* without the next line, gcc may core dump. Gcc sometimes
        gets confused if you have too many registers */

     &a; &b;

     /*printf("a %u b %u\n", a, b);*/

     /* low word */
     MUL(d7, d6);
     SWAP(d5);
     MUL(d7, d5);
     SWAP(d7);
     MUL(d7, d4);
     d4 += d5;
     SWAP(d4);
     d4 &= 0xFFFF0000;
     d4 += d6;
     return(d4);
}
#endif

#ifdef L_mulsi3
/* _mulsi3 (a, b) int a, b; { return a * b; } */

long multl(a, b)
long a, b;
{
     int sign = 0;
     long umultl();
     if ((a ^ b) < 0) sign = 1;
     if (a < 0) a = -a;
     if (b < 0) b = -b;
     /*printf("a is %d b is %d\n", a, b);*/
     if (sign) return(- umultl(a,b));
     else return(umultl(a,b));
}

#endif

#ifdef L_udivsi3
/*_udivsi3 (a, b) unsigned a, b; { return a / b; } */

/*
 this routine based on one in the PD forth package for the sun by Mitch Bradley
*/
```

```
unsigned udivl(u, v)
register unsigned long u, v;
{
     register unsigned short um, ul, vm, vl;
     unsigned long ans;
     unsigned long u1, v1;
     long i;
     long rem;

     if (v == 0) {
          /* should cause an exception condition */
          DIV(u, v);
          fprintf(stderr, "division by zero\n");
     }
     if (v > u) return(0);

     ul = u; SWAP(u); um = u;
     vl = v; SWAP(v); vm = v;
     if (vm == 0) {
          u = vl; v = um;
          DIV(u, v);
          /* note -- if you delete the next line, gcc goes into
             an infinite loop */
          if (vm) printf("result is %d\n", v);
          vm = v & 0xFFFF; /* dividend is in low word */
          v &= 0xFFFF0000; /* remainder is in high word */
          v += ul;
          DIV(vl, v);
          v &= 0xFFFF; /* dividend is in low word */
          u = vm;
          SWAP(u);
          return(v + u);
          /*ans = ((um / vl) << 16) + ((((um % vl) << 16) + ul) / vl);
          return(ans);*/
     }

     if (vl == 0) return(um / vm);
     SWAP(u); SWAP(v);
     if ( (u >> 3) < v) {
          for(i = 0; u >= v; i++, u -= v);
          /*printf("lt 8\n");*/
          return(i);
     }
     u1 = u; v1 = v;

     /* scale divisor */
     v1--;
     for(i = 0; ((unsigned) v1) >= 0xFFFF; v1 >>= 1, i++);
     if (++v1 > 0xFFFF) {
          i++; v1 >>=1;
     }
     u1 >>= i;
     /*printf("i is %d, u1 is %x, v1 is %x\n", i, u1, v1);*/
```

```
     ans = u1 / v1;
     rem = u - (ans * v);
     if (rem > v) {ans++; rem -= v; }
     if (rem > v) {printf("oops\n");}
     return(ans);
}
#endif

#ifdef L_divsi3

long divl(a, b)
long a, b;
{
     int sign = 0;
     if ((a ^ b) < 0) sign = 1;
     if (a < 0) a = -a;
     if (b < 0) b = -b;
     if (sign) return(-udivl(a,b));
     else return(udivl(a,b));
}
#endif

#ifdef L_umodsi3
_umodsi3 (a, b)
     unsigned a, b;
{
  /*return a % b;*/
  return (a - ((a/b)*b));
}
#endif

#ifdef L_modsi3
_modsi3 (a, b)
     int a, b;
{
  /*return a % b;*/
  return( a - ((a/b) * b));
}
#endif

#ifdef L_lshrsi3
_lshrsi3 (a, b)
     unsigned a, b;
{
  return a >> b;
}
#endif

#ifdef L_lshlsi3
_lshlsi3 (a, b)
     unsigned a, b;
{
  return a << b;
```

```
}
#endif

#ifdef L_ashrsi3
_ashrsi3 (a, b)
     int a, b;
{
  return a >> b;
}
#endif

#ifdef L_ashlsi3
_ashlsi3 (a, b)
     int a, b;
{
  return a << b;
}
#endif


#ifdef L_divdf3

/*double _divdf3 (a, b) double a, b; { return a / b; } */

double drecip1(f1)
double f1;
{
     struct bitdouble *bdp = &f1;
     unsigned m1, m2;
     if (bdp->exp == 0 ) return(0L);
     if (bdp->mant1 == 0L) {
          bdp->exp = 0x3ff + 0x3ff - bdp->exp;
          bdp->mant2 = 0L;
          return(f1);
     }
     bdp->exp = 0x3ff + 0x3ff - bdp->exp - 1;
     m1 = (0x00100000 + bdp->mant1) >> 5;
     m2 = (0x80000000 / m1);
     /*printf("m1 %x m2 %x\n", m1, m2);*/
     m2 <<= 5;
     m2 &= 0xFFFFF;
     /*printf("exp %x mant %x\n", bdp->exp, m2);*/
     bdp->mant1 = m2;
     bdp->mant2 = 0L;
     return(f1);
}

double drecip(f)
double f;
{
     struct bitdouble *bdp;
     double quotient, remainder;
     quotient = drecip1(f);
```

```
     remainder = 1.0 - quotient * f;
     bdp = &remainder;
     for(; bdp->exp > 0x3ca; ) {
          remainder = 1.0 - (quotient*f);
          quotient = quotient + (drecip1(f)*remainder);
     }
     return(quotient);
}


double ddiv(f1, f2)
double f1, f2;
{
     return(f1 * drecip(f2));
}
#endif

#ifdef L_muldf3
/*double _muldf3 (a, b) double a, b; { return a * b; } */
qmult(m11, m12, m21, m22, p1, p2)
unsigned long m11, m12, m21, m22, *p1, *p2;
{
     register long d2 = m11;
     register unsigned long d3 = m12, d4 = m21, d5 = m22, d6 =0, d7 = 0;
     int i;
     /* guess what happens if you delete the next line.... */
     &i;
     for (i = 0; i < 11; i++) ASL(d2, d3);
     for (i = 0; i < 9; i++) ASL(d4, d5)

     for (i = 0; i < 64; i++) {
          if (d2 < 0) { ADD(d4, d5, d6, d7);}
          ASL(d2, d3);
          ASR(d4, d5);
     }
     d2 = d6;
     d3 = d7;
     for (i = 0; i < 9; i++) ASR(d2, d3);
     *p1 = d2; *p2 = d3;
}

double dmult(f1, f2)
double f1, f2;
{
     register long d2;
     register unsigned d3, d4, d5, d6, d7;
     unsigned long p1, p2;

     struct bitdouble *bdp1 = &f1, *bdp2 = &f2;
     int exp1, exp2, i;

     exp1 = bdp1->exp; exp2 = bdp2->exp;
     /* check for zero */
```

```
     if (! exp1) return(0.0);
     if (! exp2) return(0.0);
     d2 = 0x00100000 + bdp1->mant1;
     d3 = bdp1->mant2;
     d4 = 0x00100000 + bdp2->mant1;
     d5 = bdp2->mant2;
     qmult(d2, d3, d4, d5, &p1, &p2);
     d6 = p1; d7 = p2;

     if (d6 & 0x00200000) {
          ASR(d6, d7);
          exp1++;
     }

     if (bdp1->sign == bdp2->sign) bdp1->sign = 0;
     else bdp1->sign = 1;
     bdp1->exp = exp1 + exp2 - 0x3ff;
     bdp1->mant1 = d6;
     bdp1->mant2 = d7;
     return(f1);
}
#endif

#ifdef L_negdf2
/*double _negdf2 (a) double a; { return -a; } */

double dneg(num)
double num;
{
     long *i = &num;
     *i ^= 0x80000000;
     return(num);
}
#endif

#ifdef L_adddf3
/*double _adddf3 (a, b) double a, b; { return a + b; } */

double dadd(f1, f2)
double f1, f2;
{

     register long d4, d5, d6, d7;
     struct bitdouble *bdp1 = &f1, *bdp2 = &f2;
     int exp1, exp2, sign1, sign2, howmuch, temp;

     exp1 = bdp1->exp; exp2 = bdp2->exp;

     /* check for zero */
     if (! exp1) return(f2); if (! exp2) return(f1);

     /* align them */
     if (exp1 < exp2) {
```

```
          bdp1 = &f2; bdp2 = &f1;
          exp1 = bdp1->exp; exp2 = bdp2->exp;
     }
     howmuch = exp1 - exp2;
     if (howmuch > 53) return(f1);

     d7 = bdp2->mant1 + 0x00100000;
     d6 = bdp2->mant2;

     d5 = bdp1->mant1 + 0x00100000;
     d4 = bdp1->mant2;

     for (temp = 0; temp < howmuch; temp++) ASR(d7, d6);

     /* take care of negative signs */
     if (bdp1->sign) NEG(d5, d4);
     if (bdp2->sign) NEG(d7, d6);

     ADD(d7, d6, d5, d4);

     bdp1 = &f1;

     if (d5 < 0) {
          NEG(d5, d4);
          bdp1->sign = 1;
     } else bdp1->sign = 0;

     if (d5 == 0 && d4 == 0) return(0.0);

     for(howmuch = 0; d5 >= 0; howmuch++) ASL(d5, d4);

     ASL(d5, d4);
     for (temp = 0; temp < 12; temp++) ASR(d5, d4);
     bdp1->mant1 = d5;
     bdp1->mant2 = d4;
     bdp1->exp = exp1 + 11 - howmuch;
     return(f1);
}

#endif

#ifdef L_subdf3
double
_subdf3 (a, b)
     double a, b;
{
  return a+(-b);
}
#endif

#ifdef L_cmpdf2 /*int _cmpdf2 (a, b) double a, b; { if (a > b) return 1;
  else if (a < b) return -1; return 0; } */
```

```
int dcmp(f1, f2)
double f1, f2;
{
     struct bitdouble *bdp1, *bdp2;
     unsigned int s1, s2;
     bdp1 = &f1;
     bdp2 = &f2;
     s1 = bdp1->sign;
     s2 = bdp2->sign;
     if (s1 > s2) return(-1);
     if (s1 < s2) return(1);
     /* if sign of both negative, switch them */
     if (s1 != 0) {
          bdp1 = &f1;
          bdp2 = &f2;
     }
     s1 = bdp1->exp;
     s2 = bdp2->exp;
     if (s1 > s2) return(1);
     if (s1 < s2) return(-1);
     /* same exponent -- have to compare mantissa */
     s1 = bdp1->mant1;
     s2 = bdp2->mant1;
     if (s1 > s2) return(1);
     if (s1 < s2) return(-1);
     s1 = bdp1->mant2;
     s2 = bdp2->mant2;
     if (s1 > s2) return(1);
     if (s1 < s2) return(-1);
     return(0); /* the same! */
}
#endif


#ifdef L_fixunsdfsi
/*_fixunsdfsi (a) double a; { return (unsigned int) a; } */

/* #ifdef L_fixdfsi _fixdfsi (a) double a; { return (int) a; } #endif */

unsigned long dtoul(f)
double f;
{
     struct bitdouble *bdp;
     int si, ex, mant1, mant2;
     bdp = &f;
     si = bdp->sign;
     ex = bdp->exp;
     mant1 = bdp->mant1 + 0x00100000;
     mant2 = bdp->mant2;

     /* zero value */
     if (ex == 0) return(0L);
     /* less than 1 */
```

```
     if (ex < 0x3ff) return(0L);
     /* less than 0 */
     if (si ) return(0L);
     mant1 <<= 10;
     mant1 += (mant2 >> 22);
     mant1 >>= 30 + (0x3ff - ex);
     return(mant1);
}

long dtol(f)
double f;
{
     struct bitdouble *bdp = &f;

     if (bdp->sign) {
          bdp->sign = 0;
          return( - dtoul(f));
     }
     return(dtoul(f));
}
#endif

#ifdef L_fixunsdfdi

/*double _fixunsdfdi (a) double a; { union double_di u;
  u.i[LOW] = (unsigned int) a; u.i[HIGH] = 0; return u.d; } */
#endif


#ifdef L_fixdfdi
double
_fixdfdi (a)
     double a;
{
  union double_di u;
  u.i[LOW] = (int) a;
  u.i[HIGH] = (int) a < 0 ? -1 : 0;
  return u.d;
}
#endif

#ifdef L_floatsidf
/* double _floatsidf (a) int a; { return (double) a; } */

double ltod(i)
long i;
{
     int expo, shift;
     double retval;
     struct bitdouble *bdp = &retval;
     if (i == 0) {
          long *t = &retval;
          t[0] = 0L;
```

```
          t[1] = 0L;
          return(retval);
     }
     if (i < 0) {
          bdp->sign = 1;
          i = -i;
     } else bdp->sign = 0;
     shift = i;
     for (expo = 0x3ff + 31 ; shift > 0; expo--, shift <<= 1);
     shift <<= 1;
     bdp->exp = expo;
     bdp->mant1 = shift >> 12;
     bdp->mant2 = shift << 20;
     return(retval);
}

#endif

#ifdef L_floatdidf
/* ok as is -- will call other routines */
double
_floatdidf (u)
     union double_di u;
{
  register double hi
    = ((double) u.i[HIGH]) * (double) 0x10000 * (double) 0x10000;
  register double low = (unsigned int) u.i[LOW];
  return hi + low;
}
#endif


#ifdef L_addsf3
SFVALUE
_addsf3 (a, b)
     union flt_or_int a, b;
{
  union flt_or_int intify; return INTIFY ((double) a.f + (double) b.f);
}
#endif

#ifdef L_negsf2
SFVALUE
_negsf2 (a)
     union flt_or_int a;
{
  union flt_or_int intify;
  return INTIFY (-((double) a.f));
}
#endif

#ifdef L_subsf3
SFVALUE
```

```
_subsf3 (a, b)
     union flt_or_int a, b;
{
  union flt_or_int intify;
  return INTIFY (((double) a.f - (double) b.f));
}
#endif

#ifdef L_cmpsf2
SFVALUE
_cmpsf2 (a, b)
     union flt_or_int a, b;
{
  union flt_or_int intify;
  double a1, b1;
  a1 = a.f; b1 = b.f;
  if ( a1 > b1)
    return 1;
  else if (a1 < b1)
    return -1;
  return 0;
}
#endif

#ifdef L_mulsf3
SFVALUE
_mulsf3 (a, b)
     union flt_or_int a, b;
{
  union flt_or_int intify;
  return INTIFY (((double) a.f * (double) b.f));
}
#endif

#ifdef L_divsf3
SFVALUE
_divsf3 (a, b)
     union flt_or_int a, b;
{
  union flt_or_int intify;
  return INTIFY (((double) a.f / (double) b.f));
}
#endif

#ifdef L_truncdfsf2
float dtof(d)
double d;
{
     struct bitdouble *bdp = &d;
     float retval;
     struct bitfloat *bfp = &retval;
     int tempval;
```

```
     bfp->sign = bdp->sign;
     if (bdp->exp == 0) return ((float) 0.0);
     bfp->exp = bdp->exp - 0x400 + 0x80;
     tempval = (bdp->mant1 << 4 ) + ((0xF0000000 & bdp->mant2) >> 28);
     /* round */
     tempval++;
     if (tempval == 0x01000000) bfp->exp++;
     bfp->mant = tempval >> 1;
     printf("d is %f, tempval is %x\n",d, tempval);
     printf("exp is %d\n", bfp->exp);
     return(retval);
}

SFVALUE
_truncdfsf2 (a)
     double a;
{
  union flt_or_int intify;
  return INTIFY (dtof(a));
}
#endif

#ifdef L_extendsfdf2
double ftod(f)
union flt_or_int f;
{
     double retval;
     struct bitfloat *bfp = &f.f;
     struct bitdouble *bdp = &retval;
     if (bfp->exp == 0) return(0.0);
     bdp->sign = bfp->sign;
     bdp->exp = 0x400 - 0x80 + bfp->exp;
     bdp->mant1 = bfp->mant >> 3;
     bdp->mant2 = (bfp->mant & 0x7) << 29;
     /*printf("returning %f from extendsfdf2\n", retval);*/
     return(retval);
}

double
_extendsfdf2 (a)
     union flt_or_int a;
{
  union flt_or_int intify;
  double retval;
  return (ftod(a));
}
#endif


From -3261782730546555565
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,74bea48b99d29281,start
X-Google-Attributes: gid100493,public
```

X-Google-ArrivalTime: 1988-08-24 15:35:20 PST
Path: utzoo!attcan!uunet!wyse!vsi1!ames!pasteur!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i071: break -- break out of programs with a keypress
Keywords: GST assembler
Message-ID: <4606@saturn.ucsc.edu>
Date: 24 Aug 88 22:35:20 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 345
Approved: koreth@ssyx.ucsc.edu

Submitted-by: uunet!mcvax!philmds!leo (Leo de Wit)
Posting-number: Volume 1, Issue 71
Archive-name: break

The following program, when placed in the AUTO folder or started by hand,
lets you interrupt any program (except the desktop).
The corresponding binary is sent to the moderator of comp.binaries.atari.st.
The program was assembled and linked with the assembler and linker from the
GST-C compiler.

For correspondence conceirning this program (bugs, questions etc.) try

      L. J. M. de Wit
      Nachtegaallaan 7
      5731XP Mierlo
      Holland
      e-mail: ..!mcvax!philmds!leo
              (or perhaps ..!hp4nl!philmds!leo)

------------------------------ starts here ---------------------------------
```
*****************************************************************************
*                                                                           *
*    break.asm  version 1.0 of 20 August 1988   (C) L.J.M. de Wit 1988      *
*                                                                           *
* This software may be used and distributed freely if not used commercially *
* and the originator (me) is mentioned.                                     *
*                                                                           *
*****************************************************************************
*
* NAME
*    break - stop current program on receipt of interrupt character
*
* SYNTAX
*    break [-e|-d|-z] [-i|-c<code>]
*
* DESCRIPTION
*
*    After installing break, any program can be interrupted.
*    This is achieved by 'extending' the existing keyboard interrupt
*    routine: after executing the old code the break character check
*    is done.
```

```
*
*    The various flags have the following meaning:
*       e(nable) : the current break character will end the program.
*                  the code returned is -32, what seems to be the standard
*                  value for programs interrupted by ^C in GEMDOS.
*       d(isable): no actions are done; this restores the old behaviour
*       z(ero)   : the current break character will be discarded (made 0)
*                  in the input buffer; this can be used to disable ^C.
*       i(nput)  : the break character is prompted for. Combinations with
*                  shift, control and alternate keys are also allowed.
*                  Useful for specifying the break character interactively.
*       c(ode)   : specifies the break character as a hexadecimal code.
*                  The hex code must follow the 'c' flag immediately.
*                  Useful for specifying the break character from a script.
*    Of the flags e,d and z only one should be used; e is the default.
*    Also, of the flags i and c only one should be used; control-delete is
*    the default. This is done on purpose; you can always change it to ^C
*    if you want to (or whatever key you like).
*
*    The break program can be reused indefinitely, because a next invocation
*    is not made memory resident; it only modifies parameters in the first
*    invocation (the resident one).
*
*    The program can be placed into the \AUTO folder to be installed
*    automatically, or activated 'by hand'. If placed in the \AUTO folder, it
*    should of course have a .PRG extension (break.prg); as \AUTO folder
*    programs don't get arguments, the break will be enabled and the break
*    character is control-delete in this case.
*
* BUGS/SHORTCOMINGS
*    A nice extension would be the possibility to catch the interrupt from
*    a user program; this could be achieved by using a new trap. As this
*    implies restoring the old interrupt catch routine when the program
*    exits, and maybe also core dumps could be added to the action of an
*    (other) interrupt character, such a more general signal mechanism is
*    not added (yet). Gives me time to think of a nice implementation 8-).
*
* JOKE
*    Gimme a break, huh?!
*

      module  break
      section s.ccode

* character codes
tab      equ   9
lf       equ   10
cr       equ   13

* GEMDOS & (X)BIOS stuff
gemdos   equ   1
bios     equ   13
xbios    equ   14
```

```
ptermres equ   $31
setexc   equ   5
cconws   equ   9
crawcin  equ   7
super    equ   $20
pterm0   equ   0
pterm    equ   $4c
iorec    equ   14

* divers
bpaglen  equ   $100
textlen  equ   12
datalen  equ   20
bsslen   equ   28
intrupt  equ   -32              * Code returned by interrupted programs
curproc  equ   $602c            * Contains base page start of current process

* iorec struct offsets; the buffer ptr. itself has offset 0
ibufsize equ 4
ibufhead equ 6
ibuftl   equ 8
ibuflow  equ 10
ibufhi   equ 12


brkinit
      move.l   4(sp),a0
      adda.w   #$80,a0          * a0 points to argument string start
      moveq.l  #0,d0
      move.b   (a0)+,d0         * Get string length &
      clr.b    (a0,d0.w)        * ensure '\0' termination
nxtbyt
      move.b   (a0)+,d0
      beq      nxtdone          * '\0' char is end of string
      cmp.b    #'A',d0
      blt.s    arge
      cmp.b    #'Z',d0
      bgt.s    arge
      or.b     #$20,d0          * Convert uppercase to lower
arge
      cmp.b    #'e',d0          * enable
      bne.s    argd
      st       enabled          * enabled = TRUE
      bra.s    nxtbyt
argd
      cmp.b    #'d',d0          * disable
      bne.s    argi
      sf       enabled          * enabled = FALSE
      bra.s    nxtbyt
argi
      cmp.b    #'i',d0          * input
      bne.s    argc
      move.l   a0,-(sp)
```

```
      pea.l    inpmsg
      move.w   #cconws,-(sp)
      trap     #gemdos           * Prompt for break char
      addq.l   #6,sp
      move.w   #crawcin,-(sp)
      trap     #gemdos           * Read the char
      addq.l   #2,sp
      move.l   d0,breakcode      * and store it
      pea.l    retmsg
      move.w   #cconws,-(sp)
      trap     #gemdos           * Print a cr/lf
      addq.l   #6,sp
      move.l   (sp)+,a0
      bra.s    nxtbyt
argc
      cmp.b    #'c',d0           * code of break char in hex
      bne.s    argz
      moveq.l  #0,d1             * d1 will contain the code
argcnxt
      move.b   (a0)+,d0
      bne.s    argcnz            * Not end of arg string yet
      move.l   d1,breakcode      * If end, store break char
      bra      nxtdone           * and goto end of arg string interpretation
argcnz
      cmp.b    #' ',d0           * End of number found
      beq.s    argcend
      cmp.b    #tab,d0           * this one too
      beq.s    argcend
      cmp.b    #'0',d0           * Now follows a series of tests and
      blt.s    userr             * conversions to find the hex digit's value.
      cmp.b    #'a',d0
      blt.s    argcnlc
      sub.w    #32,d0            * convert lower case to upper
argcnlc
      cmp.b    #'F',d0
      bgt.s    userr
      cmp.b    #'9',d0
      ble.s    argcnum
      cmp.b    #'A',d0
      blt.s    userr
      subq.l   #7,d0             * Make up for diff between '9' and 'A'
argcnum
      sub.b    #'0',d0           * '0' is the base: 0
      asl.l    #4,d1             * Multiply by 16
      or.b     d0,d1             * OR in the 4 low digits of d0 into d1
      bra.s    argcnxt
argcend
      move.l   d1,breakcode      * Store the break key code
      bra      nxtbyt
argz
      cmp.b    #'z',d0
      bne.s    argsep
      st       zeroed            * Set the 'zeroed' flag
```

```
      bra     nxtbyt
argsep
      cmp.b   #' ',d0           * Accept space
      beq     nxtbyt
      cmp.b   #tab,d0           * and tab
      beq     nxtbyt
      cmp.b   #'-',d0           * and hyphen as separators (not strictly)
      beq     nxtbyt
userr
      pea.l   usemsg            * If error in arg string show usage message
      move.w  #cconws,-(sp)
      trap    #gemdos
      addq.l  #6,sp
      move.w  #1,-(sp)
      move.w  #pterm,-(sp)
      trap    #gemdos           * and terminate with error status

nxtdone
      move.w  #1,-(sp)
      move.w  #iorec,-(sp)
      trap    #xbios
      addq.l  #4,sp
      move.l  d0,iop            * Save pointer to iorec struct
      move.l  #-1,-(sp)
      move.w  #$46,-(sp)
      move.w  #setexc,-(sp)
      trap    #bios
      addq.l  #8,sp
      move.l  d0,oldvec         * Save old keyboard interrupt vector
      lea.l   breakey,a0
      sub.l   a0,d0
      move.l  d0,diff           * Difference between start addr of old and new
      clr.l   -(sp)
      move.w  #super,-(sp)
      trap    #gemdos           * Supervisor mode
      addq.l  #2,sp
      move.l  d0,(sp)           * Save supervisor stack pointer on stack
      lea.l   magic,a0          * a0 points to 'magic' string in this prog
      move.l  diff,d0
      lea.l   (a0,d0.l),a1      * a1 points (perhaps) to 'magic' in old
      move.l  #(magicend-magic-1),d0 * # chars in 'magic' string minus one
chkmag
      cmp.b   (a0)+,(a1)+       * Check strings for equality
      dbne    d0,chkmag
      beq.s   mageq             * If old prog DID contain magic string; else:
magne
      move.w  #super,-(sp)
      trap    #gemdos           * Back to user mode (sp was still on stack)
      addq.l  #6,sp
      clr.l   diff              * First incarnation: will be made resident
      pea.l   breakey           * breakey will be new keyboard int. vector
      move.w  #$46,-(sp)
      move.w  #setexc,-(sp)
```

```
      trap     #bios             * Set it
      addq.l   #8,sp
      move.l   4(sp),a0          * basepage start
      move.l   #bpaglen,d0       * base page length
      add.l    textlen(a0),d0    * + text length
      add.l    datalen(a0),d0    * + data length
      add.l    bsslen(a0),d0     * + bss length
      clr.w    -(sp)             * return value: 0 for success
      move.l   d0,-(sp)          * # bytes to keep
      move.w   #ptermres,-(sp)   * keep process
      trap     #gemdos           * stops here...

mageq
      move.w   #super,-(sp)      * A next incarnation of break in this branch.
      trap     #gemdos           * Back to user mode (sp was still on stack)
      addq.l   #6,sp
      move.l   diff,d0
      lea.l    enabled,a0
      move.b   (a0),(a0,d0.l)    * Copy 'enabled' into old image
      lea.l    zeroed,a0
      move.b   (a0),(a0,d0.l)    * Do this too for 'zeroed'
      lea.l    breakcode,a0
      move.l   (a0),(a0,d0.l)    * And also for the current break char
      move.w   #pterm0,-(sp)
      trap     #gemdos           * Exits here with 0 as status.

breakey
* The new interrupt routine
      pea.l    breaketc          * Return to breaketc after executing old code
      move.w   sr,-(sp)          * This is because old code ends with 'rte'
      move.l   oldvec,-(sp)      * Push old vector onto stack
      rts                        * and jump to it
breaketc
      movem.l  d0-d1/a0-a2,-(sp)
      tst.b    enabled
      beq.s    breakend          * If not enabled do nothing
      move.l   iop,a0
      move.w   ibuftl(a0),d0
      cmp.w    ibufhead(a0),d0
      beq.s    breakend          * If empty keyboard buffer do nothing
      moveq.l  #2,d0
      move.l   curproc,a1
testbp
      tst.l    (a1)              * At end of list?
      beq.s    breakend          * Cur. prog was desktop or so; don't break
      move.l   36(a1),a1         * Back link to previous process
      dbra     d0,testbp
      moveq.l  #0,d0
      move.w   ibufhead(a0),d0   * d0 is used as index into char buffer
      move.l   breakcode,d1
      move.l   (a0),a1           * a1 points to character buffer
testbuf
      cmp.w    ibuftl(a0),d0
```

```
      beq.s    breakend          * Checked them all
      addq.l   #4,d0             * Increment index in buffer
      cmp.w    ibufsize(a0),d0   * If at buffer size
      bne.s    notatend
      moveq.l  #0,d0             * wrap it (circular buffer)
notatend
      cmp.l    (a1,d0.w),d1      * Is char at this index the break char?
      bne.s    testbuf           * No; check next
      tst.b    zeroed            * Yes; is 'zeroed' flag set?
      beq.s    notzeroed
      clr.l    (a1,d0.w)         * Make the break char in the buffer 0;
      bra.s    testbuf           * this eliminates any special meaning
notzeroed
      move.w   ibuftl(a0),ibufhead(a0) * Clear the entire buffer
      move.l   #stop,22(sp)      * Return will be to the routine 'stop'
breakend
      movem.l  (sp)+,d0-d1/a0-a2
      rte

stop
      move.w   #intrupt,-(sp)    * Code for interrupted programs
      move.w   #pterm,-(sp)
      trap     #gemdos           * and stop here

   section  s.data

iop         dc.l  0              * Pointer to iorec struct
breakcode   dc.l  $0053001f      * Default is control_delete
diff        dc.l  0              * Difference in byte distance old <-> new
oldvec      dc.l  0              * previous val of interrupt vector
enabled     dc.b  $ff            * 'enabled' is default true.
zeroed      dc.b  0              * 'zeroed' is default false.
magic       dc.b  'BREAK MARKER',0 * magic string that identifies this prog.
magicend
inpmsg      dc.b  'Input the break code by hitting the key(s) : ',0
usemsg      dc.b  'break: usage: break [-e|-d|-z] [-i|-c<code>]',cr,lf,0
retmsg      dc.b  cr,lf,0

   end


From -1460683099585909869
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,dd55d862383b1139,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-09-01 08:35:52 PST
Path: utzoo!attcan!uunet!lll-winken!lll-tis!ames!mailrus!uwmcsd1!ig!agate!saturn!
ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i072: newbreak -- New version of "break" program
Keywords: MadMac, 68000
Message-ID: <4677@saturn.ucsc.edu>
```

Date: 1 Sep 88 15:35:52 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 291
Approved: koreth@ssyx.ucsc.edu

Submitted-by: ames.arc.nasa.gov!atari!apratt (Allan Pratt)
Posting-number: Volume 1, Issue 72
Archive-name: newbreak

[This is a new version of Leo de Wit's "break" program, recently posted.
 It avoids the use of undocumented locations, so should be compatible with
 all versions of TOS. -sg]

Here is the source to your break.s, as modified by me. There is no
command line option to change the default hard-break key -- I'm just
lazy, I guess.

```
*****************************************************************************
*                                                                           *
*    break.asm  version 1.0 of 20 August 1988   (C) L.J.M. de Wit 1988      *
*                                                                           *
* This software may be used and distributed freely if not used commercially  *
* and the originator (me) is mentioned.                                     *
*                                                                           *
*****************************************************************************

* character codes
tab      equ   9
lf       equ   10
cr       equ   13

* GEMDOS & (X)BIOS stuff
gemdos   equ   1
bios     equ   13
xbios    equ   14
ptermres equ   $31
setexc   equ   5
cconws   equ   9
crawcin  equ   7
super    equ   $20
pterm0   equ   0
pterm    equ   $4c
iorec    equ   14

* divers
bpaglen  equ   $100
textlen  equ   12
datalen  equ   20
bsslen   equ   28
intrupt  equ   -32     ; Code returned by interrupted programs

* iorec struct offsets; the buffer ptr. itself has offset 0
ibufsize equ 4
```

```
ibufhead equ 6
ibuftl   equ 8
ibuflow  equ 10
ibufhi   equ 12


brkinit:
      move.l   4(sp),a0
      adda.w   #$80,a0      ; a0 points to argument string start
      moveq.l  #0,d0
      move.b   (a0)+,d0     ; Get string length &
      clr.b    (a0,d0.w)    ; ensure '\0' termination
nxtbyt:
      move.b   (a0)+,d0
      beq      nxtdone      ; '\0' char is end of string
      cmp.b    #'A',d0
      blt.s    arge
      cmp.b    #'Z',d0
      bgt.s    arge
      or.b     #$20,d0      ; Convert uppercase to lower
arge:
      cmp.b    #'e',d0      ; enable
      bne.s    argd
      st       enabled      ; enabled = TRUE
      bra.s    nxtbyt
argd:
      cmp.b    #'d',d0      ; disable
      bne.s    argi
      sf       enabled      ; enabled = FALSE
      bra.s    nxtbyt
argi:
      cmp.b    #'i',d0      ; input
      bne.s    argc
      move.l   a0,-(sp)
      pea.l    inpmsg
      move.w   #cconws,-(sp)
      trap     #gemdos      ; Prompt for break char
      addq.l   #6,sp
      move.w   #crawcin,-(sp)
      trap     #gemdos      ; Read the char
      addq.l   #2,sp
      move.l   d0,breakcode ; and store it
      pea.l    retmsg
      move.w   #cconws,-(sp)
      trap     #gemdos      ; Print a cr/lf
      addq.l   #6,sp
      move.l   (sp)+,a0
      bra.s    nxtbyt
argc:
      cmp.b    #'c',d0      ; code of break char in hex
      bne.s    argz
      moveq.l  #0,d1        ; d1 will contain the code
argcnxt:
```

```
      move.b   (a0)+,d0
      bne.s    argcnz       ; Not end of arg string yet
      move.l   d1,breakcode ; If end, store break char
      bra      nxtdone      ; and goto end of arg string interpretation
argcnz:
      cmp.b    #' ',d0      ; End of number found
      beq.s    argcend
      cmp.b    #tab,d0      ; this one too
      beq.s    argcend
      cmp.b    #'0',d0      ; Now follows a series of tests and
      blt.s    userr        ; conversions to find the hex digit's value.
      cmp.b    #'a',d0
      blt.s    argcnlc
      sub.w    #32,d0       ; convert lower case to upper
argcnlc:
      cmp.b    #'F',d0
      bgt.s    userr
      cmp.b    #'9',d0
      ble.s    argcnum
      cmp.b    #'A',d0
      blt.s    userr
      subq.l   #7,d0        ; Make up for diff between '9' and 'A'
argcnum:
      sub.b    #'0',d0      ; '0' is the base: 0
      asl.l    #4,d1        ; Multiply by 16
      or.b     d0,d1        ; OR in the 4 low digits of d0 into d1
      bra.s    argcnxt
argcend:
      move.l   d1,breakcode ; Store the break key code
      bra      nxtbyt
argz:
      cmp.b    #'z',d0
      bne.s    argsep
      st       zeroed       ; Set the 'zeroed' flag
      bra      nxtbyt
argsep:
      cmp.b    #' ',d0      ; Accept space
      beq      nxtbyt
      cmp.b    #tab,d0      ; and tab
      beq      nxtbyt
      cmp.b    #'-',d0      ; and hyphen as separators (not strictly)
      beq      nxtbyt
userr:
      pea.l    usemsg       ; If error in arg string show usage message
      move.w   #cconws,-(sp)
      trap     #gemdos
      addq.l   #6,sp
      move.w   #1,-(sp)
      move.w   #pterm,-(sp)
      trap     #gemdos      ; and terminate with error status

nxtdone:
      move.w   #1,-(sp)
```

```
      move.w   #iorec,-(sp)
      trap     #xbios
      addq.l   #4,sp
      move.l   d0,iop       ; Save pointer to iorec struct
      move.l   #-1,-(sp)
      move.w   #$46,-(sp)
      move.w   #setexc,-(sp)
      trap     #bios
      addq.l   #8,sp
      move.l   d0,oldvec    ; Save old keyboard interrupt vector
      lea.l    breakey,a0
      sub.l    a0,d0
      move.l   d0,diff      ; dist between start of old and new
      clr.l    -(sp)
      move.w   #super,-(sp)
      trap     #gemdos      ; Supervisor mode
      addq.l   #2,sp
      move.l   d0,(sp)
      lea.l    magic,a0     ; a0 points to 'magic' string in this prog
      move.l   diff,d0
      lea.l    (a0,d0.l),a1 ; a1 points (perhaps) to 'magic' in old
      move.l   #(magicend-magic-1),d0    ; # chars in 'magic' string - 1
chkmag:
      cmp.b    (a0)+,(a1)+  ; Check strings for equality
      dbne     d0,chkmag
      beq.s    mageq        ; If old prog DID contain magic string; else:
magne:
      move.w   #super,-(sp)
      trap     #gemdos      ; Back to user mode (sp was still on stack)
      addq.l   #6,sp
      clr.l    diff         ; First incarnation: will be made resident
      pea.l    breakey      ; breakey will be new keyboard int. vector
      move.w   #$46,-(sp)
      move.w   #setexc,-(sp)
      trap     #bios        ; Set it
      addq.l   #8,sp
      move.l   4(sp),a0     ; basepage start
      move.l   #bpaglen,d0  ; base page length
      add.l    textlen(a0),d0    ; + text length
      add.l    datalen(a0),d0    ; + data length
      add.l    bsslen(a0),d0     ; + bss length
      clr.w    -(sp)        ; return value: 0 for success
      move.l   d0,-(sp)     ; # bytes to keep
      move.w   #ptermres,-(sp)   ; keep process
      trap     #gemdos           ; stops here...

mageq:
      move.w   #super,-(sp) ; second run of this program
      trap     #gemdos      ; Back to user mode
      addq.l   #6,sp
      move.l   diff,d0
      lea.l    enabled,a0
      move.b   (a0),(a0,d0.l)    ; Copy 'enabled' into old image
```

```
      lea.l    zeroed,a0
      move.b   (a0),(a0,d0.l)    ; Do this too for 'zeroed'
      lea.l    breakcode,a0
      move.l   (a0),(a0,d0.l)    ; And also for the current break char
      move.w   #pterm0,-(sp)
      trap     #gemdos      ; Exits here with 0 as status.

breakey:
* The new interrupt routine
      pea.l    breaketc     ; Return to breaketc after executing old code
      move.w   sr,-(sp)     ; This is because old code ends with 'rte'
      move.l   oldvec,-(sp) ; Push old vector onto stack
      rts          ; and jump to it

breaketc:
      tst.b    enabled
      beq      dorte        ; If not enabled do nothing

      movem.l  d0-d1/a0-a2,-(sp)
      move.l   iop,a0
      move.w   ibuftl(a0),d0
      cmp.w    ibufhead(a0),d0
      beq.s    dontbreak    ; If empty keyboard buffer do nothing
      move.l   (a0),a1
      move.l   (a1,d0.w),d1 ; get the last entry
      and.l    #$00ff00ff,d1     ; mask only scan & ASCII part
      cmp.l    breakcode,d1 ; check against break code
      beq.s    normalbrk
      cmp.l    hardbreak,d1 ; check against stronger break code
      bne.s    dontbreak
      moveq.l  #-1,d1       ; make long negative - means hard break
normalbrk:
      tst.b    zeroed       ; we have a break; is 'zeroed' flag set?
      bne.s    dozeroed
      subq.w   #4,d0        ; DECREMENT d0 with wrap
      bpl.s    wrapdone
      add.w    ibufsize(a0),d0
wrapdone:
      move.w   d0,ibuftl(a0)
      bra.s    gotbreak
dozeroed:
      move.w   ibuftl(a0),ibufhead(a0)        ; Clear the entire buffer
gotbreak:
      tst.l    d1      ; hard break?
      bmi.s    dobreak
      btst.b   #5,20(sp)    ; test Super bit on stacked SR
      bne.s    dontbreak

dobreak:
      movem.l  (sp)+,d0-d1/a0-a2
      move.l   #stop,2(sp)  ; Return will be to the routine 'stop'
      rte
```

```
dontbreak:
      movem.l  (sp)+,d0-d1/a0-a2
dorte:
      rte

stop:
      move.w   #intrupt,-(sp)    ; Code for interrupted programs
      move.w   #pterm,-(sp)
      trap     #gemdos      ; and stop here

*   section  s.data

iop:        dc.l  0             ; Pointer to iorec struct
breakcode:  dc.l  $002E0000     ; Default is alt-C
hardbreak:  dc.l  $002D0000     ; Alt-X means I really mean it!
diff:       dc.l  0             ; Difference in distance old <-> new
oldvec:     dc.l  0             ; previous val of interrupt vector
enabled:    dc.b  $ff           ; 'enabled' is default true.
zeroed:     dc.b  0             ; 'zeroed' is default false.
magic:      dc.b  'BREAK MARKER',0   ; magic string identifies this prog.
magicend:
inpmsg:     dc.b  'Input the break code by hitting the key(s) : ',0
usemsg:     dc.b  'break: usage: break [-e|-d|-z] [-i|-c<code>]',cr,lf,0
retmsg:     dc.b  cr,lf,0

   end



==========================================
Opinions expressed above do not necessarily    -- Allan Pratt, Atari Corp.
reflect those of Atari Corp. or anyone else.     ...ames!atari!apratt


From 986967521488786210
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,46676d6d2b80a5cc,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-09-01 08:39:10 PST
Path: utzoo!utgpu!attcan!uunet!lll-winken!lll-tis!ames!mailrus!uwmcsd1!ig!agate!
saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i073: timecode -- Determine 68000 code speed
Keywords: Lattice
Message-ID: <4678@saturn.ucsc.edu>
Date: 1 Sep 88 15:39:10 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 240
Approved: koreth@ssyx.ucsc.edu

Submitted-by: uunet!mcvax!philmds!leo (Leo de Wit)
Posting-number: Volume 1, Issue 73
```

Archive-name: timecode

The following program, timecode, is mainly meant for the programmer crowd.
It lets you time a series of 68000 instructions (entered as short hex numbers)
which of course should constitute valid (and sensible) code; the result is
displayed as a number of clock ticks.

The corresponding uuencoded binary and a manual page have been sent to
the moderator of comp.binaries.atari.st.
Compiled with the Lattice C compiler; I used a different startup.bin to
obtain a smaller program (not included).

For correspondence conceirning this program (bugs, questions etc.) try

      L. J. M. de Wit
      Nachtegaallaan 7
      5731XP Mierlo
      Holland
      e-mail: ..!mcvax!philmds!leo
              (or perhaps ..!hp4nl!philmds!leo)

---------- S t a r t s h e r e ----------

```
/*****************************************************************************
 *                                                                           *
 *    timecode.c version 1.0 of 27 August 1988   (C) L.J.M. de Wit 1988      *
 *                                                                           *
 * This software may be used and distributed freely if not used commercially *
 * and the originator (me) is mentioned.                                     *
 *                                                                           *
 *****************************************************************************
 *
 * NAME
 *    timecode - time execution of MC68000 code
 *
 * SYNTAX
 *    timecode [-repeat] <shorthex1> <shorthex2> ...
 *
 * DESCRIPTION
 *    Timecode times the time needed to execute a series of MC68K instructions.
 *    The time taken is expressed in clock ticks.
 *    The repeat flag is optional; it is a number indicating how many times
 *    the series will be repeated (default 100000) so that timing can be
 *    accurate; a lower repeat count will fasten the execution but leads to
 *    less accuracy.
 *
 *    One of the advantages of timecode is that timings can be calculated
 *    online.
 *
 * RESTRICTIONS
 *    1) Up to 32 codes can be timed at once. This is generally no objection,
 *       since most of the time one times only one instruction at a time.
 *    2) When using opcodes that read from or write to memory the pointers
 *       used will have to be prepared correctly. A convenient way is to
```

```
 *       time the instruction to load the address register, then time both
 *       the address register load and the instruction that references
 *       memory via that pointer. Subtract the first from the second to
 *       obtain the actual number of ticks for the 'mem-ref' instruction.
 *       This whole story applies to whatever code needs proper
 *       initialization. A safe place for memory references is the screen
 *       memory.
 *    3) Take care when using branches and stuff like jsr, rts etc.
 *       Your code should handle that one way or another.
 *    4) All registers may be used except the stack pointer. SP may be
 *       used if it restored also. The items currently on the stack must
 *       be left undisturbed; if you need to access (SP), first lower SP,
 *       and increment it afterwards.
 */

/*
 *    DERIVATION OF TIMING FORMULA.
 *
 *    The formula used for the calculation of the number of ticks is
 *    (using integer division):
 *       ticks = (7791*total + (repeat / 2))/repeat - 32;
 *    where
 *       ticks : # of ticks one execution of the series takes
 *       total : # of milliseconds the complete test takes
 *       repeat: # of repetitions of the series
 *
 *    This has been derived as follows:
 *       total # of effective ticks for the test =
 *       (total - call) * eff * 8000 =
 *       repeat * (ticks + extra)
 *    where
 *       call  : # of milliseconds lost by calling the test function
 *               and saving registers. This is negligable.
 *       eff   : a coefficient (max. 1) that indicates how many ticks of
 *               the processor are really used - the others are lost in
 *               interrupt handling. Measuring gave 0.9739 as value; if the
 *               vbl semaphore was set it increased to 0.9863. The former
 *               value has been used.
 *       extra : # of ticks needed to handle each repetition.
 *               For this program its value is 32.
 *    The 8000 is in fact due to the processor's clock frequency (8Mhz);
 *    a factor 1000 can be found in the number total, which is expressed
 *    in milliseconds instead of seconds.
 *    Filling in eff and extra and adding (repeat / 2) to the nominator
 *    to achieve rounding of the integer division rather than truncating
 *    yields the formula used.
 */

/* Because stdio and memory allocation were not needed, the module was
 * linked with a different startup.bin that does not load unnecessary code.
 * In this way the program size has decreased from 12666 to 3391.
 */
```

```
#include <osbind.h>
#include <portab.h>

#define MAXARGC 32                      /* max # of codes */
#define DEF_REPEATS 100000              /* default # of repetitions */
#define HZ_200 (*(ulong *)0x4ba)        /* current 200 Hz clock value */

typedef unsigned long ulong;

static ulong do_run(), mtime();
static short hexnum();
static void prdec();
static void err_exit();

int main(argc,argv)
int argc;
char **argv;
{
   ulong total,                         /* total time for test in millisec. */
         repeat = DEF_REPEATS,          /* # of repetitions */
         ticks;                         /* ticks per repetition for code */
   int i;

   i = 1;
   if ((argc > 1) && (argv[1][0] == '-')) {
      i++; repeat = atol(argv[1] + 1);
   }
   total = do_run(argc - i, argv + i,repeat);
   ticks = (7791*total + (repeat / 2))/repeat - 32;
   prdec(ticks); Cconws(" ticks\r\n");
   Pterm0();
}

static ulong do_run(argc,argv,repeat)  /* do the test */
int argc;                              /* # of codes */
char **argv;                           /* argv contains codes as strings */
long repeat;                           /* # of repetitions */
{
   int i;
   ulong millitime;                    /* both clock and delta-time */
   short code[MAXARGC + 10],           /* will contain function to be timed */
         *cptr;                        /* pointer into code */

   if (argc > MAXARGC) {
      err_exit("too many codes",(char *)0);
   }
   cptr = code;
   *cptr++ = 0x48e7;                   /* movem.l  d0-d7/a0-a6,-(sp) */
   *cptr++ = 0xfffe;                   /* ** save regs */
   *cptr++ = 0x2f2f;                   /* move.l   64(sp),-(sp) */
   *cptr++ = 0x0040;                   /* ** push copy of repeat count */
   for (i = 0; i < argc; i++) {        /* ** now enter all codes supplied */
      *cptr++ = hexnum(argv[i]);
```

```
   }
   *cptr++ = 0x5397;                    /* subq.l   #1,(sp)   ** decr repeat */
   *cptr++ = 0x66fc - 2 * argc;         /* bne.s    again  ** jump back */
   *cptr++ = 0x588f;                    /* addq.l   #4,sp  ** pop repeat */
   *cptr++ = 0x4cdf;                    /* movem.l  (sp)+,d0-d7/a0-a6 */
   *cptr++ = 0x7fff;                    /* ** restore regs */
   *cptr++ = 0x4e75;                    /* rts   ** return; timing done */
   millitime = mtime();                 /* save current time */
   (*(void (*)())code)(repeat);         /* execute the code just build */
   return mtime() - millitime;          /* return clock difference in msec */
}

static short hexnum(str)                /* return the value of str as hex num */
char *str;
{
   short retval = 0;
   short c;
   char *s;
   int i;

   for (s = str, i = 0 ; *s != '\0'; i++) {
      if (i == 4) {                     /* max is 4 hex digits */
         err_exit("invalid hex number",str);
      }
      c = *s++;
      if ((c > '9') && (c < 'A')) {
         err_exit("invalid hex number",str);
      }
      if ((c >= 'a') && (c <= 'z')) {
         c += 'A' - 'a';
      }
      if (c >= 'A') {
         c -= 7;
      }
      c -= '0';
      if ((c < 0) || (c > 15)) {
         err_exit("invalid hex number",str);
      }
      retval = (retval << 4) + c;
   }
   return retval;
}

static void prdec(num)                  /* print a ulong in decimal */
ulong num;
{
   if (num >= 10) {                     /* recursion is nice! */
      prdec(num/10);                    /* print all but the last digit */
   }
   Cconout('0'+ (num % 10));            /* and the last one */
}

static ulong mtime()                    /* current clock time in millisec */
```

```
{
   ulong retval;
   long ssp;

   ssp = Super(0);                     /* reading the clock needs S mode */
   retval = 5 * HZ_200;                /* * 5 to yield milliseconds */
   Super(ssp);

   return retval;
}

static void err_exit(str1,str2)        /* Exit after error, printing message */
char *str1, *str2;
{
   char prbuf[80];

   strcpy(prbuf,"timecode: ");
   strcat(prbuf,str1);
   if (str2 != (char *)0) {
      strcat(prbuf,": ");
      strcat(prbuf,str2);
   }
   strcat(prbuf,"\r\n");
   Cconws(prbuf);                      /* Now print it */
   Pterm(1);                           /* and stop with error status */
}
------------------------ E n d s      h e r e -----------------------------


From 1928917751195116685
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,8296fe4be66718a5,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1988-09-26 21:24:31 PST
Path: utzoo!attcan!uunet!wyse!vsi1!ames!pasteur!agate!saturn!ssyx!koreth
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v01i074: achlib -- Memory management and other routines for MadMac
part01/02
Keywords: MadMac
Message-ID: <4945@saturn.ucsc.edu>
Date: 27 Sep 88 04:24:31 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1004
Approved: koreth@ssyx.ucsc.edu

Submitted-by: achowe@watmsg.waterloo.edu (Anthony C. Howe)
Posting-number: Volume 1, Issue 74
Archive-name: achlib/part01

#! /bin/sh
# This is a shell archive.  Remove anything before the "#! /bin/sh" line,
# then unpack it by saving it in a file and typing "sh file."
```

```
#
# Wrapped by Steven Grimm (koreth) at ssyx on Mon Sep 26 21:23:57 1988
#
# unpacks with default permissions
#
# Contents : ACHACC.S ACHAPP.S ACHSLIB.H ACHSLIB.S LIST.C MAKEFILE
#    MEMORY.C MTEST.C MTEST.LNK README RUNTIME.H
#
if `test ! -s ACHACC.S`
then
echo "x - ACHACC.S"
cat > ACHACC.S << '@\Rogue\Monster\'
;*** achacc.s
;
;    871124


;*** Long Stack Space

STACK     =         $200


;*** Accessory Start

          .text
          .extern   _main
          .even

          move.l    #U_STACK, sp
          jsr       _main
          move.w    #0, (sp)
          trap      #1


;*** Save Menu Line

          .text
_MENUSAV::
          movea.l   _scrloc, a0
          lea       MENULINE, a1
          move.w    #$1e0, d0
.1:
          move.l    (a0)+, (a1)+
          dbf       d0, .1
          rts

          .bss
          .even
MENULINE: .ds.l     $1f0


;*** Restore menu line
```

```
          .text
_MENURES::
          movea.l   _scrloc, a0
          lea       MENULINE, a1
          move.w    #$1e0, d0
.1:
          move.l    (a1)+, (a0)+
          dbf       d0, .1
          rts


;*** AES Call
;
; e  long pointer to AES parameter block

          .text
_crystal::
          move.l    4(sp), d1
          move.w    #200, d0
          trap      #2
          rts


          .bss
          .even
retsav:   .ds.l     1
          .ds.l     STACK
U_STACK:  .ds.l     1



;*** Control Array Settings
;
;
;    This .data is used to set entrys 1, 2 and 3 in the control
;    array.

        .data
        .even
_ctrl_cnts::

;    Application Manager
        .dc.b   0, 1, 0                 ; func 010
        .dc.b   2, 1, 1                 ; func 011
        .dc.b   2, 1, 1                 ; func 012
        .dc.b   0, 1, 1                 ; func 013
        .dc.b   2, 1, 1                 ; func 014
        .dc.b   1, 1, 1                 ; func 015
        .dc.b   0, 0, 0                 ; func 016
        .dc.b   0, 0, 0                 ; func 017
        .dc.b   0, 0, 0                 ; func 008
        .dc.b   0, 1, 0                 ; func 019
;    Event Manager
```

```
        .dc.b   0, 1, 0                 ; func 020
        .dc.b   3, 5, 0                 ; func 021
        .dc.b   5, 5, 0                 ; func 022
        .dc.b   0, 1, 1                 ; func 023
        .dc.b   2, 1, 0                 ; func 024
        .dc.b   16, 7, 1                ; func 025
        .dc.b   2, 1, 0                 ; func 026
        .dc.b   0, 0, 0                 ; func 027
        .dc.b   0, 0, 0                 ; func 028
        .dc.b   0, 0, 0                 ; func 009
;   Menu Manager
        .dc.b   1, 1, 1                 ; func 030
        .dc.b   2, 1, 1                 ; func 031
        .dc.b   2, 1, 1                 ; func 032
        .dc.b   2, 1, 1                 ; func 033
        .dc.b   1, 1, 2                 ; func 034
        .dc.b   1, 1, 1                 ; func 005
        .dc.b   0, 0, 0                 ; func 006
        .dc.b   0, 0, 0                 ; func 007
        .dc.b   0, 0, 0                 ; func 008
        .dc.b   0, 0, 0                 ; func 009
;   Object Manager
        .dc.b   2, 1, 1                 ; func 040
        .dc.b   1, 1, 1                 ; func 041
        .dc.b   6, 1, 1                 ; func 042
        .dc.b   4, 1, 1                 ; func 043
        .dc.b   1, 3, 1                 ; func 044
        .dc.b   2, 1, 1                 ; func 045
        .dc.b   4, 2, 1                 ; func 046
        .dc.b   8, 1, 1                 ; func 047
        .dc.b   0, 0, 0                 ; func 048
        .dc.b   0, 0, 0                 ; func 049
;   Form Manager
        .dc.b   1, 1, 1                 ; func 050
        .dc.b   9, 1, 1                 ; func 051
        .dc.b   1, 1, 1                 ; func 002
        .dc.b   1, 1, 0                 ; func 003
        .dc.b   0, 5, 1                 ; func 004
        .dc.b   0, 0, 0                 ; func 005
        .dc.b   0, 0, 0                 ; func 006
        .dc.b   0, 0, 0                 ; func 007
        .dc.b   0, 0, 0                 ; func 008
        .dc.b   0, 0, 0                 ; func 009
;   Dialog Manager
        .dc.b   0, 0, 0                 ; func 060
        .dc.b   0, 0, 0                 ; func 061
        .dc.b   0, 0, 0                 ; func 062
        .dc.b   0, 0, 0                 ; func 003
        .dc.b   0, 0, 0                 ; func 004
        .dc.b   0, 0, 0                 ; func 005
        .dc.b   0, 0, 0                 ; func 006
        .dc.b   0, 0, 0                 ; func 007
        .dc.b   0, 0, 0                 ; func 008
```

```
        .dc.b   0, 0, 0                 ; func 009
;    Graphics Manager
        .dc.b   4, 3, 0                 ; func 070
        .dc.b   8, 3, 0                 ; func 071
        .dc.b   6, 1, 0                 ; func 072
        .dc.b   8, 1, 0                 ; func 073
        .dc.b   8, 1, 0                 ; func 074
        .dc.b   4, 1, 1                 ; func 075
        .dc.b   3, 1, 1                 ; func 076
        .dc.b   0, 5, 0                 ; func 077
        .dc.b   1, 1, 1                 ; func 078
        .dc.b   0, 5, 0                 ; func 009
;    Scrap Manager
        .dc.b   0, 1, 1                 ; func 080
        .dc.b   0, 1, 1                 ; func 081
        .dc.b   0, 0, 0                 ; func 082
        .dc.b   0, 0, 0                 ; func 083
        .dc.b   0, 0, 0                 ; func 084
        .dc.b   0, 0, 0                 ; func 005
        .dc.b   0, 0, 0                 ; func 006
        .dc.b   0, 0, 0                 ; func 007
        .dc.b   0, 0, 0                 ; func 008
        .dc.b   0, 0, 0                 ; func 009
;    fseler Manager
        .dc.b   0, 2, 2                 ; func 090
        .dc.b   0, 0, 0                 ; func 091
        .dc.b   0, 0, 0                 ; func 092
        .dc.b   0, 0, 0                 ; func 003
        .dc.b   0, 0, 0                 ; func 004
        .dc.b   0, 0, 0                 ; func 005
        .dc.b   0, 0, 0                 ; func 006
        .dc.b   0, 0, 0                 ; func 007
        .dc.b   0, 0, 0                 ; func 008
        .dc.b   0, 0, 0                 ; func 009
;    Window Manager
        .dc.b   5, 1, 0                 ; func 100
        .dc.b   5, 1, 0                 ; func 101
        .dc.b   1, 1, 0                 ; func 102
        .dc.b   1, 1, 0                 ; func 103
        .dc.b   2, 5, 0                 ; func 104
        .dc.b   6, 1, 0                 ; func 105
        .dc.b   2, 1, 0                 ; func 106
        .dc.b   1, 1, 0                 ; func 107
        .dc.b   6, 5, 0                 ; func 108
        .dc.b   0, 0, 0                 ; func 009
;    Resource Manger
        .dc.b   0, 1, 1                 ; func 110
        .dc.b   0, 1, 0                 ; func 111
        .dc.b   2, 1, 0                 ; func 112
        .dc.b   2, 1, 1                 ; func 113
        .dc.b   1, 1, 1                 ; func 114
        .dc.b   0, 0, 0                 ; func 115
        .dc.b   0, 0, 0                 ; func 006
```

```
        .dc.b   0, 0, 0                 ; func 007
        .dc.b   0, 0, 0                 ; func 008
        .dc.b   0, 0, 0                 ; func 009
;    Shell Manager
        .dc.b   0, 1, 2                 ; func 120
        .dc.b   3, 1, 2                 ; func 121
        .dc.b   1, 1, 1                 ; func 122
        .dc.b   1, 1, 1                 ; func 123
        .dc.b   0, 1, 1                 ; func 124
        .dc.b   0, 1, 2                 ; func 125



@\Rogue\Monster\
else
  echo "shar: Will not over write ACHACC.S"
fi
if `test ! -s ACHAPP.S`
then
echo "x - ACHAPP.S"
cat > ACHAPP.S << '@\Rogue\Monster\'
;*** achstart.s
;
;    871121


;*** Stack & Heap Sizes
;
;    The stack sits below the heap and has STACK long words
;    allocated. The heap size (memory kept after a shrink)
;    is set by EXTRA. _brk monitors heap space r=  ests.

STACK     =         $200
EXTRA     =         $2500


;*** Basepage Structure

LOTPA     =         $00       ; Low TPA address
HITPA     =         $04       ; High TPA address
CODEBASE  =         $08       ; Code segment start
CODELEN   =         $0c       ; Code segment length
DATABASE  =         $10       ; Data segment start
DATALEN   =         $14       ; Data segment length
BSSBASE   =         $18       ; Bss  segment start
BSSLEN    =         $1c       ; Bss  segment length
DTA       =         $20       ; Disk Transfer Address
PARENT    =         $24       ; parent's basepage
RESERVE1  =         $28       ; resevered
ENVIRON   =         $2c       ; environment string
FCB2      =         $38       ; 2nd parsed fcb
FCB1      =         $5c       ; 1st parsed fcb
```

```
COMMAND   =         $80       ; Command tail


          .globl    _FUBAR

;*** Start here

          .text
          .globl    __main
          .globl    __exit
          .globl    __sovf
          .even
_FUBAR:
          move.l    4(sp),a5
          move.l    a5,__base
          move.l    #U_STACK,sp

          move.l    CODELEN(a5),d0
          add.l     DATALEN(a5),d0
          add.l     BSSLEN(a5),d0
          add.l     #EXTRA,d0

          move.l    d0,-(sp)
          move.l    a5,-(sp)
          clr.w     -(sp)
          move      #$4a,-(sp)          ; do Mshrink
          trap      #1
          add.l     #12,sp

          move.l    __base,a0           ; Load C external
          move.l    BSSBASE(a0),a1      ; a1 -> .bss region
          adda.l    BSSLEN(a0),a1       ; a1 -> 1st heap loc
          move.l    a1,__break          ; Put in "break" loc

          lea.l     COMMAND(a0),a2      ; a2 -> command line
          move.b    (a2)+,d0            ; d0 = byte count
          andi.l    #$ff,d0             ; clear junk
          move.w    d0,-(a7)            ; push length
          move.l    a2,-(a7)            ; Push commnd
          clr.l     a6                  ; Clear frame pointer
          jsr       __main
          jmp       __exit



          .globl    _brk

;*** brk

          .text
_brk:
          move.l   4(sp),a0             ; new break r=  est
```

```
          move.l    #-1,d0
          move.l   __base,a1           ; basepage address
          move.l   HITPA(a1),a1        ; get MAX break
          cmpa.l    a0,a1              ; return -1 if
          ble.s     BRKX               ;  r=  est >= TOP
          move.l    a0,__break
          clr.l     d0
BRKX:
          rts


__sovf:
          move.w    #9,d0              ; String print
          lea       ovf,a0             ; a0-> message
          move.l    a0,d1              ; load proper reg
          trap      #2                 ; Issue message
__exit:
          move.w    #0,d0              ; Exit
          trap      #2



          .globl    _blkfill

;*** Block Fill function:
;
;    blkfill (dest, char, count);
;
; e  BYTE    *dest
;    BYTE    char to fill
;    WORD    count
;
          .text
_blkfill:
          move.l    4(a7),a0       ; -> Output area
          move.w    8(a7),d1       ; =  output char
          move.w    10(a7),d0      ; =  output count
          ext.l     d0             ; make it long
          subq.l    #1,d0          ; decrement
          ble.s     filldone       ; Done if le
fillit:   move.b    d1,(a0)+       ; move a byte
          dbra      d0,fillit      ; Continue
filldone: clr.l     d0             ; always return 0
          rts



          .globl    _index
          .globl    _strchr

;*** Index function to find out if a particular
;    character is in a string.
;
```

```
; e  long pointer to string
;    short character to find
;
; x  D0.L = pointer to character else NULL if failed

          .text
_index:
_strchr:
          move.l    4(a7),a0       ; a0 -> String
          move.w    8(a7),d0       ; D0 = desired character
xindex:
          tst.b     (a0)           ; EOS?
          bne.s     notend         ; No, continue to look
          clr.l     d0             ; Not found
          rts
notend:
          cmp.b     (a0)+,d0       ; check for character
          bne.s     xindex
          move.l    a0,d0          ; Found it
          subq.l    #1,d0          ; set return pointer
          rts



          .globl    ___pname
          .globl    ___tname
          .globl    ___lname
          .globl    ___xeof

;*** Runtime Data

          .data
          .even
ovf:      .dc.b     'Stack Overflow$'   ; Error message
___pname: .dc.b     'C runtime',0       ; Program name
___tname: .dc.b     'CON:',0            ; Console name
___lname: .dc.b     'LST:',0            ; List device name
___xeof:  .dc.b     $1a                 ; Control-Z



          .globl    __base
          .globl    __break
          .globl    ___cpmrv

;*** BSS Data

          .bss
          .even
__base:   .ds.l     1              ; Base Page
__break:  .ds.l     1              ; Break function
___cpmrv: .ds.w     1              ; Last CP/M return val
```

```
          .ds.l     STACK
U_STACK:  .ds.l     1



          .globl    _crystal

;*** AES Call
;
; e  long pointer to AES parameter block

          .text
_crystal:
          move.l    4(sp),d1
          move.w    #200,d0
          trap      #2
          rts



          .globl    _control

;*** AES Arrays
;
;    short control [] =
;         {
;              op code,
;              size in words of _int_in,
;              size in words of _int_out,
;              size in longs of _addr_in,
;              size in longs of _addr_out,
;         }

          .bss
          .even
_control: .ds.w     4



          .globl    _ctrl_cnts

;*** Control Array Settings
;
;
;    This .data is used to set entrys 1, 2 and 3 in the control
;    array.

        .data
        .even
_ctrl_cnts:

;    Application Manager
        .dc.b   0, 1, 0                 ; func 010
```

```
        .dc.b   2, 1, 1                 ; func 011
        .dc.b   2, 1, 1                 ; func 012
        .dc.b   0, 1, 1                 ; func 013
        .dc.b   2, 1, 1                 ; func 014
        .dc.b   1, 1, 1                 ; func 015
        .dc.b   0, 0, 0                 ; func 016
        .dc.b   0, 0, 0                 ; func 017
        .dc.b   0, 0, 0                 ; func 008
        .dc.b   0, 1, 0                 ; func 019
;   Event Manager
        .dc.b   0, 1, 0                 ; func 020
        .dc.b   3, 5, 0                 ; func 021
        .dc.b   5, 5, 0                 ; func 022
        .dc.b   0, 1, 1                 ; func 023
        .dc.b   2, 1, 0                 ; func 024
        .dc.b   16, 7, 1                ; func 025
        .dc.b   2, 1, 0                 ; func 026
        .dc.b   0, 0, 0                 ; func 027
        .dc.b   0, 0, 0                 ; func 028
        .dc.b   0, 0, 0                 ; func 009
;   Menu Manager
        .dc.b   1, 1, 1                 ; func 030
        .dc.b   2, 1, 1                 ; func 031
        .dc.b   2, 1, 1                 ; func 032
        .dc.b   2, 1, 1                 ; func 033
        .dc.b   1, 1, 2                 ; func 034
        .dc.b   1, 1, 1                 ; func 005
        .dc.b   0, 0, 0                 ; func 006
        .dc.b   0, 0, 0                 ; func 007
        .dc.b   0, 0, 0                 ; func 008
        .dc.b   0, 0, 0                 ; func 009
;   Object Manager
        .dc.b   2, 1, 1                 ; func 040
        .dc.b   1, 1, 1                 ; func 041
        .dc.b   6, 1, 1                 ; func 042
        .dc.b   4, 1, 1                 ; func 043
        .dc.b   1, 3, 1                 ; func 044
        .dc.b   2, 1, 1                 ; func 045
        .dc.b   4, 2, 1                 ; func 046
        .dc.b   8, 1, 1                 ; func 047
        .dc.b   0, 0, 0                 ; func 048
        .dc.b   0, 0, 0                 ; func 049
;   Form Manager
        .dc.b   1, 1, 1                 ; func 050
        .dc.b   9, 1, 1                 ; func 051
        .dc.b   1, 1, 1                 ; func 002
        .dc.b   1, 1, 0                 ; func 003
        .dc.b   0, 5, 1                 ; func 004
        .dc.b   0, 0, 0                 ; func 005
        .dc.b   0, 0, 0                 ; func 006
        .dc.b   0, 0, 0                 ; func 007
        .dc.b   0, 0, 0                 ; func 008
        .dc.b   0, 0, 0                 ; func 009
```

```
;    Dialog Manager
        .dc.b   0, 0, 0                 ; func 060
        .dc.b   0, 0, 0                 ; func 061
        .dc.b   0, 0, 0                 ; func 062
        .dc.b   0, 0, 0                 ; func 003
        .dc.b   0, 0, 0                 ; func 004
        .dc.b   0, 0, 0                 ; func 005
        .dc.b   0, 0, 0                 ; func 006
        .dc.b   0, 0, 0                 ; func 007
        .dc.b   0, 0, 0                 ; func 008
        .dc.b   0, 0, 0                 ; func 009
;    Graphics Manager
        .dc.b   4, 3, 0                 ; func 070
        .dc.b   8, 3, 0                 ; func 071
        .dc.b   6, 1, 0                 ; func 072
        .dc.b   8, 1, 0                 ; func 073
        .dc.b   8, 1, 0                 ; func 074
        .dc.b   4, 1, 1                 ; func 075
        .dc.b   3, 1, 1                 ; func 076
        .dc.b   0, 5, 0                 ; func 077
        .dc.b   1, 1, 1                 ; func 078
        .dc.b   0, 5, 0                 ; func 009
;    Scrap Manager
        .dc.b   0, 1, 1                 ; func 080
        .dc.b   0, 1, 1                 ; func 081
        .dc.b   0, 0, 0                 ; func 082
        .dc.b   0, 0, 0                 ; func 083
        .dc.b   0, 0, 0                 ; func 084
        .dc.b   0, 0, 0                 ; func 005
        .dc.b   0, 0, 0                 ; func 006
        .dc.b   0, 0, 0                 ; func 007
        .dc.b   0, 0, 0                 ; func 008
        .dc.b   0, 0, 0                 ; func 009
;    fseler Manager
        .dc.b   0, 2, 2                 ; func 090
        .dc.b   0, 0, 0                 ; func 091
        .dc.b   0, 0, 0                 ; func 092
        .dc.b   0, 0, 0                 ; func 003
        .dc.b   0, 0, 0                 ; func 004
        .dc.b   0, 0, 0                 ; func 005
        .dc.b   0, 0, 0                 ; func 006
        .dc.b   0, 0, 0                 ; func 007
        .dc.b   0, 0, 0                 ; func 008
        .dc.b   0, 0, 0                 ; func 009
;    Window Manager
        .dc.b   5, 1, 0                 ; func 100
        .dc.b   5, 1, 0                 ; func 101
        .dc.b   1, 1, 0                 ; func 102
        .dc.b   1, 1, 0                 ; func 103
        .dc.b   2, 5, 0                 ; func 104
        .dc.b   6, 1, 0                 ; func 105
        .dc.b   2, 1, 0                 ; func 106
        .dc.b   1, 1, 0                 ; func 107
```

```
        .dc.b   6, 5, 0                 ; func 108
        .dc.b   0, 0, 0                 ; func 009
;    Resource Manger
        .dc.b   0, 1, 1                 ; func 110
        .dc.b   0, 1, 0                 ; func 111
        .dc.b   2, 1, 0                 ; func 112
        .dc.b   2, 1, 1                 ; func 113
        .dc.b   1, 1, 1                 ; func 114
        .dc.b   0, 0, 0                 ; func 115
        .dc.b   0, 0, 0                 ; func 006
        .dc.b   0, 0, 0                 ; func 007
        .dc.b   0, 0, 0                 ; func 008
        .dc.b   0, 0, 0                 ; func 009
;    Shell Manager
        .dc.b   0, 1, 2                 ; func 120
        .dc.b   3, 1, 2                 ; func 121
        .dc.b   1, 1, 1                 ; func 122
        .dc.b   1, 1, 1                 ; func 123
        .dc.b   0, 1, 1                 ; func 124
        .dc.b   0, 1, 2                 ; func 125



@\Rogue\Monster\
else
  echo "shar: Will not over write ACHAPP.S"
fi
if `test ! -s ACHSLIB.H`
then
echo "x - ACHSLIB.H"
cat > ACHSLIB.H << '@\Rogue\Monster\'
/*
     achslib.h

     871203    A C Howe       Implementation
     880109                   Deque stuff
*/

#define achslib_header

/* AES Exports */
extern short   gl_apid;
extern short   global [];
extern short   control [];

/* System Variables */
extern char DEFRES;
extern char MONITOR;
extern char CONTERM;

/* Anthony's Stuff */
extern char    TERM [][3];
extern short   XMAX;
```

```
extern short   YMAX;
extern short   TABLEN;
extern void    GOTOXY ();     /* short, short */

extern void memmove();
extern void   strcat();
extern void strncpy();
extern unsigned strlen();
extern void strnput();

extern char *  STRTOK ();     /* char *, char * */
extern char *  NEXTTOK;
extern void    PUTDEC ();     /* short */
extern void    PUTLONG ();    /* long */
extern void    ULTOA ();      /* long, char * */
extern long    ULD ();        /* long , long */
extern long    FINDI ();      /* char *, short */
extern long    LFINDI ();     /* char *, short */

extern long gemdos ();
extern long xbios ();


/*   Atari System Variables (supervisor only) */
#define   SHOW_SHIFT     (CONTERM |= 0x08)
#define   HIDE_SHIFT     (CONTERM &= 0xf7)


/*   Basic Terminal Functions */
#define   BEEP           (Cconws (&TERM[0]))
#define   SET_TERMINAL   (Cconws (&TERM[1]))
#define   RST_TERMINAL   (Cconws (&TERM[2]))
#define   HOME_CURSOR    (Cconws (&TERM[3]))
#define   SCREEN_CLEAR   (Cconws (&TERM[4]))
#define   EOL_CLEAR      (Cconws (&TERM[5]))
#define   EOP_CLEAR      (Cconws (&TERM[6]))
#define   ON_CURSOR      (Cconws (&TERM[7]))
#define   OFF_CURSOR     (Cconws (&TERM[8]))
#define   HI_LITE        (Cconws (&TERM[9]))
#define   LO_LITE        (Cconws (&TERM[10]))
#define   INS_LINE       (Cconws (&TERM[11]))
#define   DEL_LINE       (Cconws (&TERM[12]))
#define   UP_CURSOR      (Cconws (&TERM[13]))
#define   DOWN_CURSOR    (Cconws (&TERM[14]))
#define   RIGHT_CURSOR   (Cconws (&TERM[15]))
#define   LEFT_CURSOR    (Cconws (&TERM[16]))

#define NULL        0
#define LNULL       0L
#define BLOCK       128            /* must be powers of 2 */
#define DBLOCK      256            /* double block */
#define CTRLZ       0x1a
#define RUB         0x7f
```

```
#define FILES       0x00
#define READONLY    0x01
#define HIDDEN      0x02
#define SYSTEM      0x04
#define VOLUME      0x08
#define DIRECTORY   0x10
#define ARCHIVE     0x20


typedef union cbuffer
{
     struct template
     {
          char max_line;
          char length;
          char line [BLOCK - 2 * sizeof (char)];
     } c;
     char buf [BLOCK];
}
     CBUFFER;


typedef struct command_entry
{
     char *name;
     void (*function) ();
}
     COMMAND;


typedef struct dta_buf
{
     char dummy[21];      /* first 21 bytes not used */
     char attr;           /* file attribute */
     short ftime;         /* file time stamp */
     short fdate;         /* file date stamp */
     long fsize;          /* file size in bytes */
     char fname[14];      /* file name and extension */
}
     DTA;


typedef struct deque
{
     long *base;
     long *left;
     long *right;
     long *end;
}
     DEQUE;

#define   LEFT      0
```

```
#define   RIGHT     1

extern DEQUE * MKDEQ ();      /* long */
extern void    DEQADD ();     /* DEQUE *, short, long */
extern long    DEQDEL ();     /* DEQUE *, short */
extern void    RMDEQ ();      /* DEQUE * */


@\Rogue\Monster\
else
  echo "shar: Will not over write ACHSLIB.H"
fi
if `test ! -s ACHSLIB.S`
then
echo "x - ACHSLIB.S"
cat > ACHSLIB.S << '@\Rogue\Monster\'
;*** achslib.s
;
;    880507

;*** ASCII Code Equates

BS        ==        $08
HT        ==        $09
LF        ==        $0a
VT        ==        $0b
FF        ==        $0c
CR        ==        $0d
ZEOF      ==        $1a
ESC       ==        $1b
SPACE     ==        $20

;*** Atari ST Key Equates

;    Shift Bits
KRSHIFT   ==        $01
KLSHIFT   ==        $02
KCTRL     ==        $04
KALT      ==        $08
KCAPS     ==        $10
KRMOUSE   ==        $20
KLMOUSE   ==        $40

;    Function Keys
KF1       ==        $003b0000
KF2       ==        $003c0000
KF3       ==        $003d0000
KF4       ==        $003e0000
KF5       ==        $003f0000
KF6       ==        $00400000
KF7       ==        $00410000
KF8       ==        $00420000
KF9       ==        $00430000
```

```
KF10      ==        $00440000

KESC      ==        $0001001b
KRETURN   ==        $001c000d
KENTER    ==        $0072000d
KBS       ==        $000e0008
KDEL      ==        $0053007f

KCTLDEL   ==        $0053001f
KCTLLEFT  ==        $00730000
KCTLRIGHT ==        $00740000

KHOME     ==        $00470000
KUP       ==        $00480000
KLEFT     ==        $004b0000
KRIGHT    ==        $004d0000
KDOWN     ==        $00500000
KTAB      ==        $000f0009

KINSERT   ==        $00520000
KUNDO     ==        $00610000
KHELP     ==        $00620000

;*** GEMDOS Equates

CCONIN    ==        1    ; standard console in
CCONOUT   ==        2    ; standard console out
CAUXIN    ==        3    ; auxilary in
CAUXOUT   ==        4    ; auxilary out
CPRNOUT   ==        5    ; standard printer out
CRAWIO    ==        6    ; Raw I/O to standard I/O
CRAWIN    ==        7    ; Raw console in, no echo, no ctrl interpret
CNECIN    ==        8    ; Raw console in, no echo, ctrl interpreted
CCONWS    ==        9    ; write NULL terminated string to standard output
CCONRS    ==        10   ; read edited string from standard input
CCONIS    ==        11   ; check status of standard input


;*** Atari System Variables
;
;    NOTE: must be in Supervisor mode to modify.

_DEFRES   ==        $44a      ; default monitor resolution
_MONITOR  ==        $44c      ; monitor type
_CONTERM  ==        $484      ; attributes for the console system


;*** Terminal Definition Area

          .data
          .even
_TERM::
bell:     .dc.b     7,0,0           ; bell code string
```

```
tset:     .dc.b     0,0,0            ; terminal init

trst:     .dc.b     0,0,0            ; terminal de-init

curhome:  .dc.b     ESC,'H',0      ; cursor home

clrscr:   .dc.b     ESC,'E',0      ; clear screen

clreol:   .dc.b     ESC,'K',0      ; clear to EOL

clreop:   .dc.b     ESC,'J',0      ; clear to EOP

curon:    .dc.b     ESC,'e',0      ; cursor on

curoff:   .dc.b     ESC,'f',0      ; cursor off

hilite:   .dc.b     ESC,'p',0      ; inverse on

lolite:   .dc.b     ESC,'q',0      ; inverse off

insline:  .dc.b     ESC,'L',0      ; insert line

delline:  .dc.b     ESC,'M',0      ; delete line

curup:    .dc.b     ESC,'A',0      ; cursor up

curdn:    .dc.b     ESC,'B',0      ; cursor down

currt:    .dc.b     ESC,'C',0      ; cursor right

curlt:    .dc.b     ESC,'D',0      ; cursor left



;*** GotoXY for VT-52 terminal
;
; e  short X coord
;    short Y coord
;
; x  none
;
;    Position cursor at .text coords (X,Y).

          .data
          .even

curaddr:  .dc.b     ESC,'Y',0      ; cursor addressing lead in

_XMAX::   .dc.w     80             ; max number of columns
_YMAX::   .dc.w     25             ; max number of rows
_TABLEN:: .dc.w     4              ; tab length to use
```

```
          .text
_GOTOXY::
          link      a6,#0
          movea.l   #curaddr,a0    ; pnt to cursor addressing string
          move.l    a0,-(sp)       ; put on stack
          move.w    #CCONWS,-(sp)  ; send string to terminal
          trap      #1             ;   do GEMDOS call
          addq.l    #6,sp          ; correct stack

          move.w    _YMAX,d1       ; point to ymax value
          move.w    10(a6),d0      ; get y argument
          bsr.s     GOTOXY1

          move.w    _XMAX,d1       ; point to xmax value
          move.w    8(a6),d0       ; get x argument
          bsr.s     GOTOXY1
          unlk      a6
          rts
GOTOXY1:
          ext.l     d0
          divu      d1,d0
          swap      d0             ; q = q mod max
          addi.w    #32,d0         ; q = q + offset

          move.w    d0,-(sp)
          move.w    #CCONOUT,-(sp)
          trap      #1
          addq.l    #4,sp
          rts



;void memmove( char* dest, char* source, long length )

              .text
_memmove::
              .cargs        .dest.l, .source.l, .len.l
              movea.l       .source(sp),a0      ; get source
              movea.l       .dest(sp),a1        ; get target
              move.l        .len(sp),d0       ; get length
              beq.s         .x                      ; exit if length is zero
              cmpa.l        a1,a0                   ; check  for s < t
              blt.s         .2                      ; move tail first
.1:
              move.b        (a0)+,(a1)+             ; move source byte to target
              subq.l        #1,d0                   ; decrement length
              bne           .1                      ; loop till zero
              bra.s         .x
.2:
              adda.l        d0,a0                   ; do tail first
              adda.l        d0,a1
.3:
              move.b        -(a0),-(a1)
```

```
              subq.l        #1,d0
              bne           .3
.x:
             rts



;void strnput( char* s, unsigned n )
          .text
_strnput::
          link      a6,#0
          movem.l   d3/a3, -(sp)
          movea.l   8(a6), a3
          adda.w    12(a6), a3
          cmpa.l    8(a6), a3
          beq.s     STRNPUT1
          move.b    (a3), d3
          clr.b     (a3)
STRNPUT1:
          move.l    8(a6), -(sp)
          move.w    #CCONWS, -(sp)
          trap      #1
          addq.l    #6, sp
          cmpa.l    8(a6), a3
          beq.s     STRNPUT2
          move.b    d3, (a3)
STRNPUT2:
          movem.l   (sp)+, d3/a3
          unlk      a6
          rts



;void strncpy( char* s1, char* s2, unsigned n )

          .text
_strncpy::
          .cargs        .s1.l, .s2.l, .n.l
        movea.l         .s2(sp),a0
        movea.l    .s1(sp),a1
        move.w     .n(sp),d0
        beq.s      .X
.1:


From 4500522135661234114
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,bd4a63489c9ad4ad,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 156df7,bd4a63489c9ad4ad,start
X-Google-Attributes: gid156df7,public
X-Google-Thread: 100493,bd4a63489c9ad4ad,start
X-Google-Attributes: gid100493,public
```

X-Google-ArrivalTime: 1992-11-19 13:53:24 PST
Xref: sparky comp.sys.atari.st:16878 comp.sources.atari.st:67
comp.binaries.atari.st:201
Path: sparky!uunet!wupost!zaphod.mps.ohio-state.edu!magnus.acs.ohio-state.edu!
usenet.ins.cwru.edu!agate!stanford.edu!kronos.arc.nasa.gov!iscnvx!netcomsv!
spud.Hyperion.COM!spud.Hyperion.COM!not-for-mail
From: koreth@spud.Hyperion.COM (Steven Grimm)
Newsgroups: comp.sys.atari.st,comp.sources.atari.st,comp.binaries.atari.st
Subject: Atari sources and binaries available via anonymous ftp
Date: 19 Nov 1992 17:09:56 -0800
Organization: Hyperion, Mountain View, CA, USA
Lines: 9
Approved: koreth@spud.Hyperion.COM (Steven Grimm)
Message-ID: <1ehdt4INNn4p@spud.Hyperion.COM>
NNTP-Posting-Host: spud.hyperion.com

I'm pleased to announce that comp.sources.atari.st and comp.binaries.atari.st
archives are now available via anonymous ftp from host ftp.Hyperion.COM
(192.65.216.1). If the name doesn't resolve, try again in a little while;
the NIC just updated our NS records today.

The archive server at spud.Hyperion.COM is still up and running for
non-Internet users. Enjoy!

-Steve

From -1258215880152907436
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,6ba7fc2b720f9093,start
X-Google-Attributes: gid100493,public
X-Google-Thread: 156df7,6ba7fc2b720f9093,start
X-Google-Attributes: gid156df7,public
X-Google-ArrivalTime: 1993-01-09 22:25:53 PST
Xref: sparky comp.sources.atari.st:70 comp.binaries.atari.st:205
Path: sparky!uunet!spool.mu.edu!umn.edu!csus.edu!netcom.com!spud.Hyperion.COM!
spud.Hyperion.COM!not-for-mail
From: koreth@spud.Hyperion.COM (Steven Grimm)
Newsgroups: comp.sources.atari.st,comp.binaries.atari.st
Subject: RFD: What to do with comp.binaries.atari.st/comp.sources.atari.st?
Followup-To: comp.sys.atari.st
Date: 9 Jan 1993 21:41:14 -0800
Organization: Hyperion, Mountain View, CA, USA
Lines: 39
Approved: koreth@spud.hyperion.com (Steven Grimm)
Message-ID: <1ioctqINNnv0@spud.Hyperion.COM>
NNTP-Posting-Host: spud.hyperion.com

[Copied from comp.sys.atari.st, where followups are directed.]

I just posted "reverse" to comp.binaries.atari.st. It's the only submission
I've received for either group in nearly a month, unless you count all the
"Why isn't there anything in this group?" messages.

It seems clear to me that neither group is going to pick up much steam under my direction. When I post the submission guidelines, I get a trickle of new programs, but it dies off quickly.

Originally, this was going to be a simple call for a new moderator, someone who has the time to be a little more proactive about seeking out software to post, which is what I think would be necessary to get any decent volume. But upon further reflection, I started wondering if maybe c.b.a.s (and to a lesser extent c.s.a.s) was really all that useful in its present form.

With the steady increase in popularity of the ftp-based archive sites, many of which can also be accessed directly via E-mail (or indirectly via the various ftpmail servers on the net), does it really make sense any more to broadcast uuencoded binaries to every site on the net?

Perhaps the group should be more along the lines of gnu.announce, where new programs available for ftp (or mail-server access) are announced but not actually posted. This might cut down on what little net bandwidth the group eats; on the other hand, it makes retrieving software on a UUCP-only site slower, sometimes significantly so.

Not, mind you, that I'm opposed to the way the group works now, or to continuing as moderator, but neither of these may be the most effective way to serve the Atari ST user community, which is largely why I got into this moderation thing in the first place.

So what do people think? Should the group continue as is? Should I step down and let someone else take the helm? Should binaries continue to be posted, or should the group switch to a low-bandwidth "announce" format? If it did the latter, would it still be an appropriate member of the comp.binaries hierarchy? And what about comp.sources.atari.st?

-Steve

From 2557313106309641748
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 112d91,d7e2fe4cbd38ef6e,start
X-Google-Attributes: gid112d91,public
X-Google-Thread: 100493,d7e2fe4cbd38ef6e,start
X-Google-Attributes: gid100493,public
X-Google-Thread: 111a55,d7e2fe4cbd38ef6e,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 156df7,d7e2fe4cbd38ef6e,start
X-Google-Attributes: gid156df7,public
X-Google-ArrivalTime: 1993-04-10 01:09:56 PST
Xref: gmd.de comp.sys.atari.st:20336 alt.tv.babylon-5:1054
comp.sources.atari.st:49 comp.binaries.atari.st:156
Path: gmd.de!newsserver.jvnc.net!howland.reston.ans.net!noc.near.net!uunet!olivea!
gossip.pyramid.com!pyramid!weitek!netcomsv!spud.Hyperion.COM!spud.Hyperion.COM!
not-for-mail
From: koreth@spud.Hyperion.COM (Steven Grimm)

Newsgroups:
comp.sys.atari.st,alt.tv.babylon-5,comp.sources.atari.st,comp.binaries.atari.st
Subject: ADMIN: ftp.hyperion.com unavailable this weekend
Message-ID: <1q4h4p$or@spud.Hyperion.COM>
Date: 9 Apr 93 19:00:41 GMT
Organization: Hyperion, Mountain View, CA, USA
Lines: 11
Approved: koreth@spud.hyperion.com
NNTP-Posting-Host: spud.hyperion.com

The ftp.hyperion.com ftp site (which houses the official comp.sources.atari.st/
comp.binaries.atari.st archives and a lot of Babylon 5 files, thus the
crossposting of this message) will be unavailable until the afternoon of
Monday, April 12. Apparently the ftp server we're using has a nasty security
hole. I won't have time to install the fixed version until Monday, so in the
meantime I'm being paranoid. For Atari users, archive-server@spud.hyperion.com
will not be affected by this shutdown.

Sorry for the inconvenience.

-Steve

From 4689926305139392162
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,8ba628d93eed28f0,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,8ba628d93eed28f0,start
X-Google-Attributes: gid100493,public
X-Google-Thread: 156df7,8ba628d93eed28f0,start
X-Google-Attributes: gid156df7,public
X-Google-ArrivalTime: 1993-11-20 10:01:40 PST
Xref: gmd.de comp.binaries.atari.st:222 comp.sources.atari.st:59
comp.sys.atari.st:31108
Path: gmd.de!xlink.net!sol.ctr.columbia.edu!spool.mu.edu!umn.edu!csus.edu!
netcom.com!netcomsv!spud.Hyperion.COM!spud.Hyperion.COM!not-for-mail
From: koreth@spud.Hyperion.COM (Steven Grimm)
Newsgroups: comp.binaries.atari.st,comp.sources.atari.st,comp.sys.atari.st
Subject: ADMIN: Seeking new moderator for comp.{binaries,sources}.atari.st
Followup-To: comp.sys.atari.st
Date: 20 Nov 1993 08:52:04 -0800
Organization: Hyperion, Mountain View, CA, USA
Lines: 32
Approved: koreth@spud.Hyperion.COM (Steven Grimm)
Message-ID: <2clhvk$ipf@spud.Hyperion.COM>
NNTP-Posting-Host: spud.hyperion.com

I think it's time for a new moderator for the sources and binaries groups.

For some time now, as everyone is no doubt aware, I haven't been able to
devote as much time to the groups as they deserve. It looks like that'll
continue into the future, and it's not fair to all of you that I continue
to hang on to moderatorship when someone more active could be making the

groups go somewhere.

So: I'm now soliciting volunteers to take over the groups. I think it's best if one person moderates both, simply because most of the stuff in comp.sources.atari.st arrives packaged in with binaries, but of course I'll consider offers to take over just one group.

Requirements:

- Willingness to devote several hours a week, or more, to the groups.
- Some understanding of how Usenet works from a technical standpoint, enough to modify and maintain the posting scripts and other tools.
- Enough disk space to handle the occasional multi-megabyte submission.
- Net access that won't go away (i.e. no students who lose their accounts or don't have access to them over the summer.)

Internet access is a plus, but not required; if someone without Internet access takes the job, naturally, outside arrangements will have to be made if the archives are to be available via ftp.

Please reply to koreth@hyperion.com if you're interested in the job.

Thanks to everyone who's participated in the group over the years. It's been fun.

-Steve

From -59334445678030114
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,4e6d5d02e52c06ea,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: f9437,4e6d5d02e52c06ea,start
X-Google-Attributes: gidf9437,public
X-Google-Thread: 100493,4e6d5d02e52c06ea,start
X-Google-Attributes: gid100493,public
X-Google-Thread: 156df7,4e6d5d02e52c06ea,start
X-Google-Attributes: gid156df7,public
X-Google-ArrivalTime: 1993-12-01 02:51:22 PST
Xref: gmd.de news.groups:44281 comp.binaries.atari.st:223 comp.sources.atari.st:60 comp.sys.atari.st:31670
Path: gmd.de!newsserver.jvnc.net!howland.reston.ans.net!cs.utexas.edu!uunet!elroy.jpl.nasa.gov!decwrl!netcomsv!netcom.com!netcomsv!spud.Hyperion.COM!spud.Hyperion.COM!not-for-mail
From: koreth@spud.Hyperion.COM (Steven Grimm)
Newsgroups: news.groups,comp.binaries.atari.st,comp.sources.atari.st,comp.sys.atari.st
Subject: ADMIN: Remove comp.binaries.atari.st and comp.sources.atari.st?
Followup-To: news.groups
Date: 30 Nov 1993 22:27:42 -0800
Organization: Hyperion, Mountain View, CA, USA
Lines: 22
Approved: koreth@spud.Hyperion.COM (Steven Grimm)

Message-ID: <2dhdgu$4uu@spud.Hyperion.COM>
NNTP-Posting-Host: spud.hyperion.com

About a week and a half ago, I posted an article to comp.binaries.atari.st
and comp.sources.atari.st asking for volunteers to replace me as moderator.
One person asked for more information, then never replied to my mail, so I
can only assume he's not interested.

Which makes me wonder: is it time to kill off both groups?

Even when I have time to moderate it properly, traffic is a trickle, and
the various Atari ftp sites get far more submissions than either group.
With the availability of mail servers to fetch things from those sites,
I'm not sure there's a reason for the groups any more. Certainly the
fact that nobody else wants to moderate them is evidence that interest
is fairly low.

If the groups aren't removed, I'm not going to have time to moderate them,
so they will sit idle. I think it's dumb to have groups that nobody can
post to. Removing them entirely seems like a far better solution.

So, I leave it to you to discuss. What should be done? Followups directed
to news.groups, where this sort of thing is supposed to be talked about.

-Steve

From 5576887998194425475
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: f9437,2b840f4ffc99dd1,start
X-Google-Attributes: gidf9437,public
X-Google-Thread: 100493,2b840f4ffc99dd1,start
X-Google-Attributes: gid100493,public
X-Google-Thread: 111a55,2b840f4ffc99dd1,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 156df7,2b840f4ffc99dd1,start
X-Google-Attributes: gid156df7,public
X-Google-ArrivalTime: 1994-01-07 13:51:37 PST
Xref: gmd.de comp.sources.atari.st:61 comp.binaries.atari.st:224
comp.sys.atari.st:33659 news.groups:46706
Path: gmd.de!newsserver.jvnc.net!howland.reston.ans.net!pipex!doc.ic.ac.uk!decwrl!
decwrl!netcomsv!netcomsv!spud.Hyperion.COM!spud.Hyperion.COM!not-for-mail
From: koreth@spud.Hyperion.COM (Steven Grimm)
Newsgroups:
comp.sources.atari.st,comp.binaries.atari.st,comp.sys.atari.st,news.groups
Subject: ADMIN: New moderator for comp.binaries.atari.st, comp.sources.atari.st
Followup-To: comp.sys.atari.st
Date: 7 Jan 1994 08:48:50 -0800
Organization: Hyperion, Mountain View, CA, USA
Lines: 22
Approved: koreth@spud.Hyperion.COM (Steven Grimm)
Message-ID: <2gk3pi$qi1@spud.Hyperion.COM>
NNTP-Posting-Host: spud.hyperion.com

After I posted a while ago asking if the groups should be dissolved, several people stepped forward volunteering to moderate. I'm happy to report that one of them is just about ready to start, so as of now I'm stepping down as moderator.

Please welcome Annius Groenink, avg@cwi.nl, to both groups. Under enthusiastic new management, I hope the groups will take on new life. I'll continue to hang around both groups in an advisory capacity for a while, but this is probably the last posting you'll see from me here.

All new submissions should be sent to the abovementioned E-mail address, and you should adjust your moderators or mailpaths file accordingly until the change is reflected in the List of Moderators maintained at Uunet. The archives will remain at ftp.hyperion.com until I can find another home for them (any volunteers?)

I'd like to thank everyone who participated over the years, both readers and submitters. It's been an interesting ride.

See you all on the net!

-Steve

From 1555777734115885326
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,2cc221cb8f5c266d,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 156df7,2cc221cb8f5c266d,start
X-Google-Attributes: gid156df7,public
X-Google-Thread: 100493,2cc221cb8f5c266d,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1994-01-14 10:18:23 PST
Xref: gmd.de comp.binaries.atari.st:225 comp.sources.atari.st:62
comp.sys.atari.st:34127
Newsgroups: comp.binaries.atari.st,comp.sources.atari.st,comp.sys.atari.st
Path: gmd.de!xlink.net!howland.reston.ans.net!EU.net!sun4nl!cwi.nl!avg
From: avg@cwi.nl (Annius Groenink)
Subject: (info) A note from the new sources/binaries moderator
Message-ID: <CJMp6A.MK2@cwi.nl>
Originator: avg@mijt.cwi.nl
Sender: news@cwi.nl (The Daily Dross)
Nntp-Posting-Host: mijt.cwi.nl
Organization: CWI, Amsterdam
Date: Fri, 14 Jan 1994 16:44:34 GMT
Approved: avg@cwi.nl
Expires: Sat, 14 May 1994 16:44:34 GMT
Lines: 55

Hello everybody.

As Steven Grimm decided to devote his time to other matters, I will from now on be the moderator for the comp.binaries.atari.st and comp.sources.atari.st groups. You will not see many changes. The posting conventions have not changed more than slightly. It is as yet unclear what is going to happen to the archives Steven kept. Therefore you should be aware that it may not be possible to retrieve earlier postings when they have expired.

Please take note of the following issues:

- Submissions to both the binaries and sources groups should now be sent to the following e-mail address:

  avg@cwi.nl

- As ftp archives are becoming a popular alternative for publication of sources and binaries, I will encourage the submission of announcements for uploads to ftp archives. Archive maintainers, feel free to include a note in your uploading instructions encouraging people to submit a short description of their publications to my e-mail address.

- Although I certainly try to do things as well as Steven, I may of course make mistakes in the beginning. In that case, sorry for any inconvenience. Never hesitate to let me know if something is not quite the way you would like to see it.

- For those who have submitted articles to the old addresses, I am working on those. I will post one submission to both groups today, to see if everything works out well. The others will follow soon. If I haven't contacted you about your submission, then either it's still in examination, or it looks OK and it will soon be posted.

Of course, I always appreciate any comments or ideas on how to improve the usefulness of the groups.

I think we'd all like to thank Steven Grimm for the great work he has been doing for the groups. I am also endebted to him for his advice and tools for maintaining the groups.

Thank you all very much and enjoy,

Annius Groenink, Moderator. (avg@cwi.nl)

From -8766000927988576869
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,8d36fce7bd6e3f88,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1994-01-14 10:27:00 PST
Newsgroups: comp.sources.atari.st
Path: gmd.de!xlink.net!howland.reston.ans.net!math.ohio-state.edu!usc!
elroy.jpl.nasa.gov!swrinde!cs.utexas.edu!uunet!EU.net!sun4nl!cwi.nl!avg
From: euapol@eua.ericsson.se (Tommy Pollak)
Subject: v07i001: FrontStrip -- Strip Leading Blanks in Simula
Message-ID: <csas.7.1@cwi.nl>
Originator: avg@mijt.cwi.nl
Sender: news@cwi.nl (The Daily Dross)
Nntp-Posting-Host: mijt.cwi.nl
Organization: CWI, Amsterdam
Date: Fri, 14 Jan 1994 16:47:43 GMT
Approved: avg@cwi.nl
Lines: 58

Submitted-by: euapol@eua.ericsson.se (Tommy Pollak)
Environment: Simula (OK on Falcon030/4MB, not on 520ST)
Posting-number: Volume 7, Issue 1
Archive-name: frontstr.txt

Issue: comp.sources.atari.st, Volume 7 Number 1

Name: FrontStrip - Simula

Description: A Simula Procedure which strips leading blanks off a text.

Configuration: See below.

Simula is one of the first object-oriented programming languages. It enhances Algol 60 with classes and other features. For the Atari there is a port of the Lund Simula system by the Dortmund University in Germany.

The Simula system did not fit my old 520 ST, but works OK on my Falcon030 (4/64 MB).

I would like to share a piece of experience with you.

In the swedish book "Objektorienterad programmering och algoritmer i Simula" by Serafim Dahl and Kjell Lindqvist, Studentlitteratur 1993, a text procedure to strip the leading blanks of a text is defined.

The procedure will not give correct results for texts with exactly one

non-blank character. Below comes a modified version.

```
text procedure FrontStrip (t); text t;
  begin
  t.setpos (1);
  while t.more and then t.getchar = ' ' do;
  FrontStrip :- if t.more then
              t.sub (t.Pos-1, t.length-t.pos+2)
         else
           ! next 3 lines added;
           if t.length > 0 then
                t.sub (t.Pos-1, t.length-t.pos+2)
           else
             Notext
  end of FrontStrip;
```

Tommy Pollak at ELLEMTEL, department FN, this time not speaking for the
company
Postal address: Box 1505
                S-125 05 ALVSJO
                Sweden
Phone: +46 8 727 4056   Fax: +46 8 727 41 68

From 9221139338390126150
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,31ec851b49f56709,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,31ec851b49f56709,start
X-Google-Attributes: gid100493,public
X-Google-Thread: 156df7,31ec851b49f56709,start
X-Google-Attributes: gid156df7,public
X-Google-ArrivalTime: 1994-01-14 10:50:29 PST
Xref: gmd.de comp.binaries.atari.st:230 comp.sources.atari.st:64
comp.sys.atari.st:34129
Newsgroups: comp.binaries.atari.st,comp.sources.atari.st,comp.sys.atari.st
Path: gmd.de!xlink.net!howland.reston.ans.net!EU.net!sun4nl!cwi.nl!avg
From: avg@cwi.nl (Annius Groenink)
Subject: A note from the new sources/binaries moderator
Message-ID: <CJMqxy.55@cwi.nl>
Originator: avg@mijt.cwi.nl
Sender: news@cwi.nl (The Daily Dross)
Nntp-Posting-Host: mijt.cwi.nl
Organization: CWI, Amsterdam
Date: Fri, 14 Jan 1994 17:22:45 GMT
Approved: avg@cwi.nl
Expires: Sat, 14 May 1994 17:22:45 GMT
Lines: 60

Hello everybody.

As Steven Grimm decided to devote his time to other matters, I will from now on be the moderator for the comp.binaries.atari.st and comp.sources.atari.st groups. You will not see many changes. The posting conventions have not changed more than slightly. It is as yet unclear what is going to happen to the archives Steven kept. Therefore you should be aware that it may not be possible to retrieve earlier postings when they have expired.

Please take note of the following issues:

- Submissions to both the binaries and sources groups should now be sent to the following e-mail address:

  avg@cwi.nl

- As ftp archives are becoming a popular alternative for publication of sources and binaries, I will encourage the submission of announcements for uploads to ftp archives. Archive maintainers, feel free to include a note in your uploading instructions encouraging people to submit a short description of their publications to my e-mail address.

- Although I certainly try to do things as well as Steven, I may of course make mistakes in the beginning. In that case, sorry for any inconvenience. Never hesitate to let me know if something is not quite the way you would like to see it.

- For those who have submitted articles to the old addresses, I am working on those. I will post one submission to both groups today, to see if everything works out well. The others will follow soon. If I haven't contacted you about your submission, then either it's still in examination, or it looks OK and it will soon be posted.

Of course, I always appreciate any comments or ideas on how to improve the usefulness of the groups.

I think we'd all like to thank Steven Grimm for the great work he has been doing for the groups. I am also endebted to him for his advice and tools for maintaining the groups.

Thank you all very much and enjoy,

Annius Groenink, Moderator. (avg@cwi.nl)

--
Annius V. Groenink | E-mail: avg@cwi.nl | Private & ZFC:
CWI, Kruislaan 413 | Room: M233 | P.O. Box 799
1098 SJ Amsterdam | Ext: 4077 | NL 3500 AT Utrecht
Netherland | Phone: +31 20 592 4077 | Phone: +31 30 803740

From 7063963992985078499
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,d312096acea618a3,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,d312096acea618a3,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1994-01-14 11:06:19 PST
Xref: gmd.de comp.sources.atari.st:65 comp.sys.atari.st:34133
Newsgroups: comp.sources.atari.st,comp.sys.atari.st
Path: gmd.de!xlink.net!howland.reston.ans.net!EU.net!sun4nl!cwi.nl!avg
From: avg@cwi.nl (Annius Groenink)
Subject: v07inf001: Introduction to comp.sources.atari.st
Message-ID: <CJMr5q.AI@cwi.nl>
Originator: avg@mijt.cwi.nl
Sender: news@cwi.nl (The Daily Dross)
Nntp-Posting-Host: mijt.cwi.nl
Organization: CWI, Amsterdam
Date: Fri, 14 Jan 1994 17:27:26 GMT
Approved: avg@cwi.nl (Annius Groenink)
Expires: Sat, 14 May 1994 17:27:26 GMT
Lines: 168

Submitted-by: avg@cwi.nl (Annius Groenink)
Posting-number: Volume 7, Info 1

This article is posted regularly. It gives a short introduction to comp.sources.atari.st and describes how to submit sources or digressions to the group.

I am always looking for suggestions on how to improve the usefulness of the newsgroup, and can be contacted at the following internet address:

   avg@cwi.nl

Annius Groenink, Moderator.

ABOUT comp.sources.atari.st

The moderated usenet group comp.sources.atari.st contains publications or announcements of sources and texts about programming for Atari TOS computers. The publications on comp.sources.atari.st may be freely

distributed as long as modifications are documented and the original authors are mentioned.

THIS IS A MODERATED GROUP. YOU CANNOT POST TO IT BY YOURSELF. Read the instructions below for submitting material intended for publication on comp.sources.atari.st.

SUBMITTING SOURCES OR DIGRESSIONS FOR PUBLICATION

Items intended for posting or queries and problem notes should be sent to avg@cwi.nl. If you are on a UUCP-only site, you can send them to uunet!cwi.nl!avg.

If you want verification of arrival, say so at the beginning of your submission, or in a cover note. I try to verify that a program works, and if I can't get it to work, I may hold up posting it for a couple of days. Please note that, except in rare cases, software without documentation will not be published. The backlog from receipt to posting varies from one to four weeks depending mostly on the set of submissions currently in my queue.

If you are submitting both sources and binaries, PLEASE send the two separately. If I have to separate your sources from your binaries by hand, your submission will most likely sit on the back burner for a while, or be rejected outright.

Sources can be submitted in ASCII form, but for large pacages, a ZOO 2.1 archive encoded using Dumas uuencode is preferred.

Please include a description of your program at the top of your article. Look at old issues of comp.binaries.atari.st for examples of the sorts of descriptions I'm looking for. In general you should indicate what the program does, on what types of Atari TOS computers, whether your program requires a certain amount of on-board memory, a hard disk or a certain screen resolution. I can't check your submission in all possible configurations, and see what works and what doesn't. Without a description, your submission is likely to be rejected. If necessary, I may put the description through a text formatter or make minor additions in order to make the postings to the group more uniform. I will, of course, never change the actual contents of a submission.

THE STRUCTURE OF COMP.SOURCES.ATARI.ST ARTICLES

There are three types of articles in comp.binaries.atari.st: "information postings", "sources" and "announcements". The source postings are also called "Issues". The type of a posting can be distinguished by the subject line:

```
  Subject: v02inf1: Introduction to comp.sources.atari.st
```

This first word in the title identifies this as the first info posting of volume two. Similarly, the subject line shown below:

```
  Subject: v06i081: godsrc -- Universe simulator
```

identifies this as the 81st source article in Volume 6. Large sources are broken up into smaller pieces, and have subject lines that look like this:

```
  Subject: v06i088: decbustr -- VMS clone in ST-LOGO, Part03/08
```

Announcements of large source packages uploaded to ftp archives are also considered for publication. The subject header for such files looks like

```
  Subject: v07ann007: GCC 7.9.4 -- Same as 7.9.3, but considerably slower
```

Certain information about the system configuration required to use the program is given on the keywords line.

```
  Keywords: MWC, Megamax, 1meg, high
```

This means that the program compiles with Mark Williams or Megamax C, requires at least one meg of RAM and only runs in high resolution. Following is a list of keywords; new ones may be added as needed. They are mostly self- explanatory.

```
  Alcyon  - Alcyon C (included with Atari developer's kit)
  MWC         - Mark Williams C
  Megamax - Megamax C
  Laser       - Laser C
  Lattice - Lattice C
  C       - Any C compiler
  OSS         - OSS Personal Pascal
  BASIC       - ST BASIC (this is not a joke!)
  uuencode    - program is uuencoded (uudecode required to unpack)
  arc         - program is archived (arc required to unpack)
  zoo-high    - program is archived (Zoo 2.1 required to unpack)
  lharc       - program is archived (lharc required to unpack)
  high        - high resolution
  medium  - medium resolution
  low         - low resolution
  1meg        - needs 1 meg of RAM
```

The References: line, if present, points to the previous part of a multi-part program, so that newsreaders (trn, nn, etc.) which can follow threads will recognize a large program as a single thread.

The first few lines of an article are auxiliary headers that look like this:

Submitted-by: david@bdt.UUCP (David Beckmeyer)
Posting-number: Volume 6, Issue 80
Archive-name: micro-rtx

The "Submitted by" is the author of the program. If you have comments about the sources published in comp.sources.atari.st this is the person to contact. When possible, this address is in domain form, otherwise it is a UUCP bang path relative to some major (backbone) site.

The second line repeats the volume/issue information for the aide of notes sites and automatic archiving programs.

The Archive-name is the "official" name of this source in the archive. Large postings will have names that look like this:

Archive-name: desktop/part01

REPORTING AND TRACKING BUGS AND PATCHES TO POSTINGS

Most of the time, bugs and updates will be posted to comp.sys.atari.st. Sometimes, "official" updates from the author will be posted submitted to comp.sources.atari.st so that they are archived with the sources.

Since there is no easy way to apply source code patches on the ST, changes to a program often result in a whole new version being posted.

To report bugs, contact the person listed in the Submitted-to header. Often there is a contact address in a README file, too. I do not maintain the sources I moderate, so don't send your bug reports to me.

--
Annius V. Groenink | E-mail: avg@cwi.nl | Private & ZFC:
CWI, Kruislaan 413 | Room: M233 | P.O. Box 799
1098 SJ Amsterdam | Ext: 4077 | NL 3500 AT Utrecht
Netherland | Phone: +31 20 592 4077 | Phone: +31 30 803740

From 1131504728753061097
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,f054f356809f7eda,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1994-01-19 12:34:05 PST
Newsgroups: comp.sources.atari.st
Path: gmd.de!xlink.net!howland.reston.ans.net!EU.net!sun4nl!cwi.nl!avg
From: C.J.Ridd@computer-science.birmingham.ac.uk
Subject: v07i002: keytbl - Produce a Key Table for your OS
Message-ID: <csas.7.2@cwi.nl>
Originator: avg@mijt.cwi.nl
Sender: news@cwi.nl (The Daily Dross)

Nntp-Posting-Host: mijt.cwi.nl
Organization: CWI, Amsterdam
Date: Wed, 19 Jan 1994 16:23:04 GMT
Approved: avg@cwi.nl
Lines: 303

Submitted-by: C.J.Ridd@computer-science.birmingham.ac.uk
Environment: ANSI C, needs dos.c with countre code identifiers
Posting-number: Volume 7, Issue 2
Archive-name: keytbl/source.part01

Issue: comp.sources.atari.st, Volume 7 number 2

Part: 1/1

Name: keytbl.c

Description: A program which produces the operating system's key tables in 3-column format.

NOTE from the moderator. If you are using an operating system different than UK, please try this program and submit its output. We are planning to compile the key tables for all languages into a comprehensive document describing the use of key codes in GEM applications.

This is C source code for 'keytbl', submitted by Chris Ridd from the Computer Science dept. at the univ. of Birmingham, United Kingdom.

As readers of usenet may have noticed, there has been a discussion on comp.sys.atari.st.tech about key combinations on foreign keyboards. One of the conclusions of this discussion was that it would be nice if we could collect the key tables for as many operating systems as we can get hold of.

'keytbl' generates an ASCII file 'keytbl.tbl' showing your TOS version, the nationality of your operating system and three columns representing the values in the keyboard table for unshifted, shifted and caps-locked keys, respectively.

The binary 'keytbl.ttp' has been posted to comp.binaries.atari.st.

If you are using a TOS in an exotic language (I haven't got the US tables yet, for instance :-) feel free to send the output of keytbl on your computer to the moderator of comp.sources.atari.st:

avg@cwi.nl

In a few weeks, I will compile the results will into a document together

with more general information as well as abstracts from the discussion
on comp.sys.atari.st.tech.

AtDhVaAnNkCsE

   Annius V. Groenink (moderator)

```
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of shell archive."
# Contents:  keytbl_s/keytbl.c keytbl_s/keytbl.tbl
# Wrapped by avg@mijt.cwi.nl on Wed Jan 19 17:13:52 1994
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'keytbl_s/keytbl.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'keytbl_s/keytbl.c'\"
else
echo shar: Extracting \"'keytbl_s/keytbl.c'\" \(1891 characters\)
sed "s/^X//" >'keytbl_s/keytbl.c' <<'END_OF_FILE'
X/*
X * Dump Xbios keyboard tables
X *
X * Corrected table size from 256 to 128
X *
X */
X
X#include <osbind.h>
X#include <dos.h>
X#include <stdio.h>
X#include <stdlib.h>
X
XFILE *fp;
X
Xvoid dumpEntry(int c)
X{
X    fprintf(fp, "0x%02x ", c);
X    if (c >= 32 && c <= 126)
X         fprintf(fp, "\"%c\"", c);
X    else
X         fprintf(fp,"   ");
X}
X
Xvoid dump(char *s, char *u, char *c)
X{
X    int i;
X    const char *l;
X
X    fprintf(fp, "TOS %d.%02x for ", _tos >> 8, _tos & 0xff);
```

```
X    switch (_country) {
X    case USA:l = "USA";                        break;
X    case FRG:l = "Germany";                    break;
X    case FRA:l = "France";                break;
X    case GBR:l = "UK";                    break;
X    case SPA:l = "Spain";                 break;
X    case ITA:l = "Italy";                 break;
X    case SWE:l = "Sweden";                break;
X    case SWF:l = "Switzerland (French)"; break;
X    case SWG:l = "Switzerland (German)"; break;
X    case TUR:l = "Turkey";                break;
X    case FIN:l = "Finland";                    break;
X    case NOR:l = "Norway";                break;
X    case DEN:l = "Denmark";                    break;
X    case SAU:l = "Saudi Arabia";               break;
X    case HOL:l = "Holland";                    break;
X    case CZE:l = "Czechoslovakia";       break;
X    case HUN:l = "Hungary";                    break;
X    default: l = NULL;                    break;
X    }
X    if (l)
X         fprintf(fp, "%s\n", l);
X    else
X         fprintf(fp, "unknown country 0x%02x\n\n", _country);
X    fprintf(fp, "\tunshift\tshift\tcapslock\n");
X    for (i = 0; i < 128; i++) {
X         fprintf(fp, "%3d =\t", i);
X         dumpEntry(s[i]);
X         fputc('\t', fp);
X         dumpEntry(u[i]);
X         fputc('\t', fp);
X         dumpEntry(c[i]);
X         fputc('\n', fp);
X    }
X
X}
X
Xint _main(int argc, char **argv)
X{
X    KEYTAB *kt;
X    const char *prog = "keytbl";
X
X    if (argv[0] && argv[0][0]) prog = argv[0];
X
X    if ((fp = fopen("keytbl.tbl", "w")) == NULL) {
X         fprintf(stderr, "%s: error creating output\n", prog);
X         return 1;
X    }
X
X    kt = (KEYTAB *)Keytbl((char *)-1, (char *)-1, (char *)-1);
X    dump(kt->shift, kt->unshift, kt->capslock);
X    fclose(fp);
X    return 0;
```

```
X}
END_OF_FILE
if test 1891 -ne `wc -c <'keytbl_s/keytbl.c'`; then
    echo shar: \"'keytbl_s/keytbl.c'\" unpacked with wrong size!
fi
# end of 'keytbl_s/keytbl.c'
fi
if test -f 'keytbl_s/keytbl.tbl' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'keytbl_s/keytbl.tbl'\"
else
echo shar: Extracting \"'keytbl_s/keytbl.tbl'\" \(4394 characters\)
sed "s/^X//" >'keytbl_s/keytbl.tbl' <<'END_OF_FILE'
XTOS 3.06 for UK
X    unshift  shift    capslock
X  0 =   0x00     0x00     0x00
X  1 =   0x1b     0x1b     0x1b
X  2 =   0x21 "!" 0x31 "1" 0x31 "1"
X  3 =   0x22 """ 0x32 "2" 0x32 "2"
X  4 =   0x9c     0x33 "3" 0x33 "3"
X  5 =   0x24 "$" 0x34 "4" 0x34 "4"
X  6 =   0x25 "%" 0x35 "5" 0x35 "5"
X  7 =   0x5e "^" 0x36 "6" 0x36 "6"
X  8 =   0x26 "&" 0x37 "7" 0x37 "7"
X  9 =   0x2a "*" 0x38 "8" 0x38 "8"
X 10 =   0x28 "(" 0x39 "9" 0x39 "9"
X 11 =   0x29 ")" 0x30 "0" 0x30 "0"
X 12 =   0x5f "_" 0x2d "-" 0x2d "-"
X 13 =   0x2b "+" 0x3d "=" 0x3d "="
X 14 =   0x08     0x08     0x08
X 15 =   0x09     0x09     0x09
X 16 =   0x51 "Q" 0x71 "q" 0x51 "Q"
X 17 =   0x57 "W" 0x77 "w" 0x57 "W"
X 18 =   0x45 "E" 0x65 "e" 0x45 "E"
X 19 =   0x52 "R" 0x72 "r" 0x52 "R"
X 20 =   0x54 "T" 0x74 "t" 0x54 "T"
X 21 =   0x59 "Y" 0x79 "y" 0x59 "Y"
X 22 =   0x55 "U" 0x75 "u" 0x55 "U"
X 23 =   0x49 "I" 0x69 "i" 0x49 "I"
X 24 =   0x4f "O" 0x6f "o" 0x4f "O"
X 25 =   0x50 "P" 0x70 "p" 0x50 "P"
X 26 =   0x7b "{" 0x5b "[" 0x5b "["
X 27 =   0x7d "}" 0x5d "]" 0x5d "]"
X 28 =   0x0d     0x0d     0x0d
X 29 =   0x00     0x00     0x00
X 30 =   0x41 "A" 0x61 "a" 0x41 "A"
X 31 =   0x53 "S" 0x73 "s" 0x53 "S"
X 32 =   0x44 "D" 0x64 "d" 0x44 "D"
X 33 =   0x46 "F" 0x66 "f" 0x46 "F"
X 34 =   0x47 "G" 0x67 "g" 0x47 "G"
X 35 =   0x48 "H" 0x68 "h" 0x48 "H"
X 36 =   0x4a "J" 0x6a "j" 0x4a "J"
X 37 =   0x4b "K" 0x6b "k" 0x4b "K"
X 38 =   0x4c "L" 0x6c "l" 0x4c "L"
```

```
X 39 =   0x3a ":" 0x3b ";" 0x3b ";"
X 40 =   0x40 "@" 0x27 "'" 0x27 "'"
X 41 =   0xff     0x60 "`" 0x60 "`"
X 42 =   0x00     0x00     0x00
X 43 =   0x7e "~" 0x23 "#" 0x23 "#"
X 44 =   0x5a "Z" 0x7a "z" 0x5a "Z"
X 45 =   0x58 "X" 0x78 "x" 0x58 "X"
X 46 =   0x43 "C" 0x63 "c" 0x43 "C"
X 47 =   0x56 "V" 0x76 "v" 0x56 "V"
X 48 =   0x42 "B" 0x62 "b" 0x42 "B"
X 49 =   0x4e "N" 0x6e "n" 0x4e "N"
X 50 =   0x4d "M" 0x6d "m" 0x4d "M"
X 51 =   0x3c "<" 0x2c "," 0x2c ","
X 52 =   0x3e ">" 0x2e "." 0x2e "."
X 53 =   0x3f "?" 0x2f "/" 0x2f "/"
X 54 =   0x00     0x00     0x00
X 55 =   0x00     0x00     0x00
X 56 =   0x00     0x00     0x00
X 57 =   0x20 " " 0x20 " " 0x20 " "
X 58 =   0x00     0x00     0x00
X 59 =   0x00     0x00     0x00
X 60 =   0x00     0x00     0x00
X 61 =   0x00     0x00     0x00
X 62 =   0x00     0x00     0x00
X 63 =   0x00     0x00     0x00
X 64 =   0x00     0x00     0x00
X 65 =   0x00     0x00     0x00
X 66 =   0x00     0x00     0x00
X 67 =   0x00     0x00     0x00
X 68 =   0x00     0x00     0x00
X 69 =   0x00     0x00     0x00
X 70 =   0x00     0x00     0x00
X 71 =   0x37 "7" 0x00     0x00
X 72 =   0x38 "8" 0x00     0x00
X 73 =   0x00     0x00     0x00
X 74 =   0x2d "-" 0x2d "-" 0x2d "-"
X 75 =   0x34 "4" 0x00     0x00
X 76 =   0x00     0x00     0x00
X 77 =   0x36 "6" 0x00     0x00
X 78 =   0x2b "+" 0x2b "+" 0x2b "+"
X 79 =   0x00     0x00     0x00
X 80 =   0x32 "2" 0x00     0x00
X 81 =   0x00     0x00     0x00
X 82 =   0x30 "0" 0x00     0x00
X 83 =   0x7f     0x7f     0x7f
X 84 =   0x00     0x00     0x00
X 85 =   0x00     0x00     0x00
X 86 =   0x00     0x00     0x00
X 87 =   0x00     0x00     0x00
X 88 =   0x00     0x00     0x00
X 89 =   0x00     0x00     0x00
X 90 =   0x00     0x00     0x00
X 91 =   0x00     0x00     0x00
```

```
X 92 =   0x00     0x00     0x00
X 93 =   0x00     0x00     0x00
X 94 =   0x00     0x00     0x00
X 95 =   0x00     0x00     0x00
X 96 =   0x7c "|" 0x5c "\" 0x5c "\"
X 97 =   0x00     0x00     0x00
X 98 =   0x00     0x00     0x00
X 99 =   0x28 "(" 0x28 "(" 0x28 "("
X100 =   0x29 ")" 0x29 ")" 0x29 ")"
X101 =   0x2f "/" 0x2f "/" 0x2f "/"
X102 =   0x2a "*" 0x2a "*" 0x2a "*"
X103 =   0x37 "7" 0x37 "7" 0x37 "7"
X104 =   0x38 "8" 0x38 "8" 0x38 "8"
X105 =   0x39 "9" 0x39 "9" 0x39 "9"
X106 =   0x34 "4" 0x34 "4" 0x34 "4"
X107 =   0x35 "5" 0x35 "5" 0x35 "5"
X108 =   0x36 "6" 0x36 "6" 0x36 "6"
X109 =   0x31 "1" 0x31 "1" 0x31 "1"
X110 =   0x32 "2" 0x32 "2" 0x32 "2"
X111 =   0x33 "3" 0x33 "3" 0x33 "3"
X112 =   0x30 "0" 0x30 "0" 0x30 "0"
X113 =   0x2e "." 0x2e "." 0x2e "."
X114 =   0x0d     0x0d     0x0d
X115 =   0x00     0x00     0x00
X116 =   0x00     0x00     0x00
X117 =   0x00     0x00     0x00
X118 =   0x00     0x00     0x00
X119 =   0x00     0x00     0x00
X120 =   0x00     0x00     0x00
X121 =   0x00     0x00     0x00
X122 =   0x00     0x00     0x00
X123 =   0x00     0x00     0x00
X124 =   0x00     0x00     0x00
X125 =   0x00     0x00     0x00
X126 =   0x00     0x00     0x00
X127 =   0x00     0x00     0x00
END_OF_FILE
if test 4394 -ne `wc -c <'keytbl_s/keytbl.tbl'`; then
    echo shar: \"'keytbl_s/keytbl.tbl'\" unpacked with wrong size!
fi
# end of 'keytbl_s/keytbl.tbl'
fi
echo shar: End of shell archive.
exit 0


From 7574481933456538045
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,ae345087628cef4c,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1994-01-19 12:34:22 PST
Newsgroups: comp.sources.atari.st
Path: gmd.de!xlink.net!howland.reston.ans.net!EU.net!sun4nl!cwi.nl!avg
```

From: jaszcz@donald.EEAP.CWRU.Edu
Subject: v07i003: mandelgen - simple mandelbrot fractal in GFA
Message-ID: <csas.20.7.3@cwi.nl>
Originator: avg@mijt.cwi.nl
Sender: news@cwi.nl (The Daily Dross)
Nntp-Posting-Host: mijt.cwi.nl
Organization: CWI, Amsterdam
Date: Wed, 19 Jan 1994 16:29:23 GMT
Approved: avg@cwi.nl
Lines: 119

Submitted-by: jaszcz@donald.EEAP.CWRU.Edu
Environment: GFA Basic 3.x
Posting-number: Volume 7, Issue 3
Archive-name: mandelgen/part01

Issue: comp.sources.atari.st, Volume 7 number 3

Part: 1/1

Name: mandelgen

Description: Mandelbrot Generator in simple (GFA Basic) terms

Configuration: GFA Basic 3.x

---

reading some of the docs on fractals, i've discovered that people cannot state the matter in simple terms, much less include code. since GFA is almost like pseudo code, i've decided to write a simple program so that everybody can understand.

as it stands, the program uses low ST resolution, but you can change it to others in minutes.

----cut----

```
'
' simple mandelbrot fractal generator
' in GFA Basic 3.x
' you can use the main program as a pseudo code for a faster
' implementation or whatever.  set_pal and col_rotate just make the final
' result more dynamic.  i hate static pictures.
'
'   --Bart
'
' jaszcz@snowhite.eeap.cwru.edu
'
DIM c%(16)
DIM col&(16)
pause%=100
```

```
set_pal
x=-2
y=-1.25
FOR yc%=0 TO 99
  FOR xc%=0 TO 319
    i&=0
    za=0
    zb=0
    '
    ' you can change this to while za < 2 and zb < 2 and i&<15
    ' it will produce a cooler image.  but the square root is the "standard"
    '
    WHILE SQR((za*za)+(zb*zb))<2 AND i&<15
      oldza=za
      oldzb=zb
      za=(oldza*oldza)-(oldzb*oldzb)+x
      zb=(2*oldza*oldzb)+y
      i&=i&+1
    WEND
    COLOR c%((i&+1) MOD 16)
    PLOT xc%,yc%
    PLOT xc%,199-yc%
    x=x+0.008125
  NEXT xc%
  y=y+0.0125
  x=-2
NEXT yc%
col_rotate
> PROCEDURE col_rotate
  x%=1
  DO
    z%=x%
    FOR y%=1 TO 14
      SETCOLOR y%,col&(z%)
      INC z%
      IF z%>14
        z%=1
      ENDIF
    NEXT y%
    INC x%
    IF x%>14
      x%=1
    ENDIF
    FOR wait%=0 TO pause%
      IF GEMDOS(11)
        key%=INP(2)
        SELECT key%
        CASE 200
          pause%=pause%+5
        CASE 208
          pause%=pause%-5
        CASE 27
          SETCOLOR 0,7,7,7
```

```
          SETCOLOR 15,0,0,0
          END
        ENDSELECT
      ENDIF
    NEXT wait%
  LOOP
RETURN
> PROCEDURE set_pal
  FOR x&=0 TO 7
    SETCOLOR x&,x&
    col&(x&)=x&
  NEXT x&
  FOR x&=7 TO 0 STEP -1
    SETCOLOR x&+8,(7-x&)*256
    col&(x&+8)=(7-x&)*256
  NEXT x&
  FOR x&=0 TO 15
    READ c%(x&)
  NEXT x&
RETURN
' vdi pen to hardware register conversion table
DATA 0,2,3,6,4,7,5,8,9,10,11,14,12,15,13,1
'
```

```
From 2161194247351372176
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,c64c066af293f5c,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1994-02-17 08:54:31 PST
Newsgroups: comp.sources.atari.st
Path: gmd.de!newsserver.jvnc.net!netnews.upenn.edu!dsinc!spool.mu.edu!
howland.reston.ans.net!EU.net!sun4nl!cwi.nl!avg
From: hjy001!rjh@relay.nl.net (Rich Hearn)
Subject: v07i004: tos1.0us.tbl -- Keyboard Table
Message-ID: <csas.7.4@cwi.nl>
Originator: avg@mijt.cwi.nl
Sender: news@cwi.nl (The Daily Dross)
Nntp-Posting-Host: mijt.cwi.nl
Organization: CWI, Amsterdam
Date: Thu, 17 Feb 1994 16:54:31 GMT
Approved: avg@cwi.nl
Lines: 158

Submitted-by: hjy001!rjh@relay.nl.net (Rich Hearn)
Posting-number: Volume 7, Issue 4
Archive-name: tables/tos1.0us.tbl

------------------------------------------------------------------------

Note from the moderator.
```

Thanks to everyone who sent in key tables. Unfortunately, the response was disappointing. We have a US, UK, Swiss and German versions. Remember, it's never too late to send in a key table! Especially French and Spanish tables would be welcome!

Annius.

TOS 1.00 for USA

	unshift	shift	capslock
0 =	0x00	0x00	0x00
1 =	0x1b	0x1b	0x1b
2 =	0x21 "!"	0x31 "1"	0x31 "1"
3 =	0x40 "@"	0x32 "2"	0x32 "2"
4 =	0x23 "#"	0x33 "3"	0x33 "3"
5 =	0x24 "$"	0x34 "4"	0x34 "4"
6 =	0x25 "%"	0x35 "5"	0x35 "5"
7 =	0x5e "^"	0x36 "6"	0x36 "6"
8 =	0x26 "&"	0x37 "7"	0x37 "7"
9 =	0x2a "*"	0x38 "8"	0x38 "8"
10 =	0x28 "("	0x39 "9"	0x39 "9"
11 =	0x29 ")"	0x30 "0"	0x30 "0"
12 =	0x5f "_"	0x2d "-"	0x2d "-"
13 =	0x2b "+"	0x3d "="	0x3d "="
14 =	0x08	0x08	0x08
15 =	0x09	0x09	0x09
16 =	0x51 "Q"	0x71 "q"	0x51 "Q"
17 =	0x57 "W"	0x77 "w"	0x57 "W"
18 =	0x45 "E"	0x65 "e"	0x45 "E"
19 =	0x52 "R"	0x72 "r"	0x52 "R"
20 =	0x54 "T"	0x74 "t"	0x54 "T"
21 =	0x59 "Y"	0x79 "y"	0x59 "Y"
22 =	0x55 "U"	0x75 "u"	0x55 "U"
23 =	0x49 "I"	0x69 "i"	0x49 "I"
24 =	0x4f "O"	0x6f "o"	0x4f "O"
25 =	0x50 "P"	0x70 "p"	0x50 "P"
26 =	0x7b "{"	0x5b "["	0x5b "["
27 =	0x7d "}"	0x5d "]"	0x5d "]"
28 =	0x0d	0x0d	0x0d
29 =	0x00	0x00	0x00
30 =	0x41 "A"	0x61 "a"	0x41 "A"
31 =	0x53 "S"	0x73 "s"	0x53 "S"
32 =	0x44 "D"	0x64 "d"	0x44 "D"
33 =	0x46 "F"	0x66 "f"	0x46 "F"
34 =	0x47 "G"	0x67 "g"	0x47 "G"
35 =	0x48 "H"	0x68 "h"	0x48 "H"
36 =	0x4a "J"	0x6a "j"	0x4a "J"
37 =	0x4b "K"	0x6b "k"	0x4b "K"
38 =	0x4c "L"	0x6c "l"	0x4c "L"
39 =	0x3a ":"	0x3b ";"	0x3b ";"
40 =	0x22 """	0x27 "'"	0x27 "'"

```
41 =    0x7e "~" 0x60 "`" 0x60 "`"
42 =    0x00     0x00     0x00
43 =    0x7c "|" 0x5c "\" 0x5c "\"
44 =    0x5a "Z" 0x7a "z" 0x5a "Z"
45 =    0x58 "X" 0x78 "x" 0x58 "X"
46 =    0x43 "C" 0x63 "c" 0x43 "C"
47 =    0x56 "V" 0x76 "v" 0x56 "V"
48 =    0x42 "B" 0x62 "b" 0x42 "B"
49 =    0x4e "N" 0x6e "n" 0x4e "N"
50 =    0x4d "M" 0x6d "m" 0x4d "M"
51 =    0x3c "<" 0x2c "," 0x2c ","
52 =    0x3e ">" 0x2e "." 0x2e "."
53 =    0x3f "?" 0x2f "/" 0x2f "/"
54 =    0x00     0x00     0x00
55 =    0x00     0x00     0x00
56 =    0x00     0x00     0x00
57 =    0x20 " " 0x20 " " 0x20 " "
58 =    0x00     0x00     0x00
59 =    0x00     0x00     0x00
60 =    0x00     0x00     0x00
61 =    0x00     0x00     0x00
62 =    0x00     0x00     0x00
63 =    0x00     0x00     0x00
64 =    0x00     0x00     0x00
65 =    0x00     0x00     0x00
66 =    0x00     0x00     0x00
67 =    0x00     0x00     0x00
68 =    0x00     0x00     0x00
69 =    0x00     0x00     0x00
70 =    0x00     0x00     0x00
71 =    0x37 "7" 0x00     0x00
72 =    0x38 "8" 0x00     0x00
73 =    0x00     0x00     0x00
74 =    0x2d "-" 0x2d "-" 0x2d "-"
75 =    0x34 "4" 0x00     0x00
76 =    0x00     0x00     0x00
77 =    0x36 "6" 0x00     0x00
78 =    0x2b "+" 0x2b "+" 0x2b "+"
79 =    0x00     0x00     0x00
80 =    0x32 "2" 0x00     0x00
81 =    0x00     0x00     0x00
82 =    0x30 "0" 0x00     0x00
83 =    0x7f     0x7f     0x7f
84 =    0x00     0x00     0x00
85 =    0x00     0x00     0x00
86 =    0x00     0x00     0x00
87 =    0x00     0x00     0x00
88 =    0x00     0x00     0x00
89 =    0x00     0x00     0x00
90 =    0x00     0x00     0x00
91 =    0x00     0x00     0x00
92 =    0x00     0x00     0x00
93 =    0x00     0x00     0x00
```

```
 94 =    0x00     0x00     0x00
 95 =    0x00     0x00     0x00
 96 =    0x00     0x00     0x00
 97 =    0x00     0x00     0x00
 98 =    0x00     0x00     0x00
 99 =    0x28 "(" 0x28 "(" 0x28 "("
100 =    0x29 ")" 0x29 ")" 0x29 ")"
101 =    0x2f "/" 0x2f "/" 0x2f "/"
102 =    0x2a "*" 0x2a "*" 0x2a "*"
103 =    0x37 "7" 0x37 "7" 0x37 "7"
104 =    0x38 "8" 0x38 "8" 0x38 "8"
105 =    0x39 "9" 0x39 "9" 0x39 "9"
106 =    0x34 "4" 0x34 "4" 0x34 "4"
107 =    0x35 "5" 0x35 "5" 0x35 "5"
108 =    0x36 "6" 0x36 "6" 0x36 "6"
109 =    0x31 "1" 0x31 "1" 0x31 "1"
110 =    0x32 "2" 0x32 "2" 0x32 "2"
111 =    0x33 "3" 0x33 "3" 0x33 "3"
112 =    0x30 "0" 0x30 "0" 0x30 "0"
113 =    0x2e "." 0x2e "." 0x2e "."
114 =    0x0d     0x0d     0x0d
115 =    0x00     0x00     0x00
116 =    0x00     0x00     0x00
117 =    0x00     0x00     0x00
118 =    0x00     0x00     0x00
119 =    0x00     0x00     0x00
120 =    0x00     0x00     0x00
121 =    0x00     0x00     0x00
122 =    0x00     0x00     0x00
123 =    0x00     0x00     0x00
124 =    0x00     0x00     0x00
125 =    0x00     0x00     0x00
126 =    0x00     0x00     0x00
127 =    0x00     0x00     0x00

--
================================================================================
! Rich Hearn               | UUCP:   decwrl,sun,uunet!uvm-gen.uucp!hjy001!rjh !
! Org: A Step In The Right Direction  | Internet:  hjy001!rjh@griffin.uvm.edu !
! "How can we know our best speaks well?" | Snail Mail: RFD 1  Box 1396       !
!  answered the Sensei; "Time will tell." |             Moretown, VT 05660 USA!
================================================================================
```

```
From 3397469053338555569
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,1824836982d64287,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1994-02-17 08:55:55 PST
Newsgroups: comp.sources.atari.st
Path: gmd.de!newsserver.jvnc.net!netnews.upenn.edu!dsinc!spool.mu.edu!
```

howland.reston.ans.net!EU.net!sun4nl!cwi.nl!avg
From: fany@fons.nbg.sub.org (Martin Sluka)
Subject: v07i005: tos3.06ger.tbl -- Keyboard Table
Message-ID: <csas.7.5@cwi.nl>
Originator: avg@mijt.cwi.nl
Sender: news@cwi.nl (The Daily Dross)
Nntp-Posting-Host: mijt.cwi.nl
Organization: CWI, Amsterdam
Date: Thu, 17 Feb 1994 16:55:55 GMT
Approved: avg@cwi.nl
Lines: 160

Submitted-by: fany@fons.nbg.sub.org (Martin Sluka)
Posting-number: Volume 7, Issue 5
Archive-name: tables/tos3.6ger.tbl

Note from the moderator.

Thanks to everyone who sent in key tables. Unfortunately, the response was disappointing. We have a US, UK, Swiss and German versions. Remember, it's never too late to send in a key table! Especially French and Spanish tables would be welcome!

Also thanks to the one who sent me the 1.4 German TOS table. It is identical to this 3.06 output.

Annius.

TOS 3.06 for Germany

	unshift	shift	capslock
0 =	0x00	0x00	0x00
1 =	0x1b	0x1b	0x1b
2 =	0x21 "!"	0x31 "1"	0x31 "1"
3 =	0x22 """	0x32 "2"	0x32 "2"
4 =	0xdd	0x33 "3"	0x33 "3"
5 =	0x24 "$"	0x34 "4"	0x34 "4"
6 =	0x25 "%"	0x35 "5"	0x35 "5"
7 =	0x26 "&"	0x36 "6"	0x36 "6"
8 =	0x2f "/"	0x37 "7"	0x37 "7"
9 =	0x28 "("	0x38 "8"	0x38 "8"
10 =	0x29 ")"	0x39 "9"	0x39 "9"
11 =	0x3d "="	0x30 "0"	0x30 "0"
12 =	0x3f "?"	0x9e	0x9e
13 =	0x60 "`"	0x27 "'"	0x27 "'"
14 =	0x08	0x08	0x08
15 =	0x09	0x09	0x09
16 =	0x51 "Q"	0x71 "q"	0x51 "Q"
17 =	0x57 "W"	0x77 "w"	0x57 "W"

```
18 =    0x45 "E" 0x65 "e" 0x45 "E"
19 =    0x52 "R" 0x72 "r" 0x52 "R"
20 =    0x54 "T" 0x74 "t" 0x54 "T"
21 =    0x5a "Z" 0x7a "z" 0x5a "Z"
22 =    0x55 "U" 0x75 "u" 0x55 "U"
23 =    0x49 "I" 0x69 "i" 0x49 "I"
24 =    0x4f "O" 0x6f "o" 0x4f "O"
25 =    0x50 "P" 0x70 "p" 0x50 "P"
26 =    0x9a     0x81     0x9a
27 =    0x2a "*" 0x2b "+" 0x2b "+"
28 =    0x0d     0x0d     0x0d
29 =    0x00     0x00     0x00
30 =    0x41 "A" 0x61 "a" 0x41 "A"
31 =    0x53 "S" 0x73 "s" 0x53 "S"
32 =    0x44 "D" 0x64 "d" 0x44 "D"
33 =    0x46 "F" 0x66 "f" 0x46 "F"
34 =    0x47 "G" 0x67 "g" 0x47 "G"
35 =    0x48 "H" 0x68 "h" 0x48 "H"
36 =    0x4a "J" 0x6a "j" 0x4a "J"
37 =    0x4b "K" 0x6b "k" 0x4b "K"
38 =    0x4c "L" 0x6c "l" 0x4c "L"
39 =    0x99     0x94     0x99
40 =    0x8e     0x84     0x8e
41 =    0x5e "^" 0x23 "#" 0x23 "#"
42 =    0x00     0x00     0x00
43 =    0x7c "|" 0x7e "~" 0x7e "~"
44 =    0x59 "Y" 0x79 "y" 0x59 "Y"
45 =    0x58 "X" 0x78 "x" 0x58 "X"
46 =    0x43 "C" 0x63 "c" 0x43 "C"
47 =    0x56 "V" 0x76 "v" 0x56 "V"
48 =    0x42 "B" 0x62 "b" 0x42 "B"
49 =    0x4e "N" 0x6e "n" 0x4e "N"
50 =    0x4d "M" 0x6d "m" 0x4d "M"
51 =    0x3b ";" 0x2c "," 0x2c ","
52 =    0x3a ":" 0x2e "." 0x2e "."
53 =    0x5f "_" 0x2d "-" 0x2d "-"
54 =    0x00     0x00     0x00
55 =    0x00     0x00     0x00
56 =    0x00     0x00     0x00
57 =    0x20 " " 0x20 " " 0x20 " "
58 =    0x00     0x00     0x00
59 =    0x00     0x00     0x00
60 =    0x00     0x00     0x00
61 =    0x00     0x00     0x00
62 =    0x00     0x00     0x00
63 =    0x00     0x00     0x00
64 =    0x00     0x00     0x00
65 =    0x00     0x00     0x00
66 =    0x00     0x00     0x00
67 =    0x00     0x00     0x00
68 =    0x00     0x00     0x00
69 =    0x00     0x00     0x00
70 =    0x00     0x00     0x00
```

```
 71 =    0x37 "7" 0x00     0x00
 72 =    0x38 "8" 0x00     0x00
 73 =    0x00     0x00     0x00
 74 =    0x2d "-" 0x2d "-" 0x2d "-"
 75 =    0x34 "4" 0x00     0x00
 76 =    0x00     0x00     0x00
 77 =    0x36 "6" 0x00     0x00
 78 =    0x2b "+" 0x2b "+" 0x2b "+"
 79 =    0x00     0x00     0x00
 80 =    0x32 "2" 0x00     0x00
 81 =    0x00     0x00     0x00
 82 =    0x30 "0" 0x00     0x00
 83 =    0x7f     0x7f     0x7f
 84 =    0x00     0x00     0x00
 85 =    0x00     0x00     0x00
 86 =    0x00     0x00     0x00
 87 =    0x00     0x00     0x00
 88 =    0x00     0x00     0x00
 89 =    0x00     0x00     0x00
 90 =    0x00     0x00     0x00
 91 =    0x00     0x00     0x00
 92 =    0x00     0x00     0x00
 93 =    0x00     0x00     0x00
 94 =    0x00     0x00     0x00
 95 =    0x00     0x00     0x00
 96 =    0x3e ">" 0x3c "<" 0x3c "<"
 97 =    0x00     0x00     0x00
 98 =    0x00     0x00     0x00
 99 =    0x28 "(" 0x28 "(" 0x28 "("
100 =    0x29 ")" 0x29 ")" 0x29 ")"
101 =    0x2f "/" 0x2f "/" 0x2f "/"
102 =    0x2a "*" 0x2a "*" 0x2a "*"
103 =    0x37 "7" 0x37 "7" 0x37 "7"
104 =    0x38 "8" 0x38 "8" 0x38 "8"
105 =    0x39 "9" 0x39 "9" 0x39 "9"
106 =    0x34 "4" 0x34 "4" 0x34 "4"
107 =    0x35 "5" 0x35 "5" 0x35 "5"
108 =    0x36 "6" 0x36 "6" 0x36 "6"
109 =    0x31 "1" 0x31 "1" 0x31 "1"
110 =    0x32 "2" 0x32 "2" 0x32 "2"
111 =    0x33 "3" 0x33 "3" 0x33 "3"
112 =    0x30 "0" 0x30 "0" 0x30 "0"
113 =    0x2e "." 0x2e "." 0x2e "."
114 =    0x0d     0x0d     0x0d
115 =    0x00     0x00     0x00
116 =    0x00     0x00     0x00
117 =    0x00     0x00     0x00
118 =    0x00     0x00     0x00
119 =    0x00     0x00     0x00
120 =    0x00     0x00     0x00
121 =    0x00     0x00     0x00
122 =    0x00     0x00     0x00
123 =    0x00     0x00     0x00
```

```
124 =    0x00     0x00     0x00
125 =    0x00     0x00     0x00
126 =    0x00     0x00     0x00
127 =    0x00     0x00     0x00
--
```


```
_____________________________________________________________________
Pirckheimerstr. 1 /   Martin "fany" Sluka   /   Breite Strasse 3
D-91207 Lauf     /------------------------/ D-90552 Roethenbach
+49-9123-75900  /   fany@fons.nbg.sub.org /      +49-911-577664
+49-9123-13874 / m.sluka@msn.zer.sub.org /   fax:+49-911-5700318
---------------------------------------------------------------------
```


```
  "But now you can kiss my ass goodbye!" -- Frederick Bulsara



From -2231835574056153643
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,5ea0469889dfa17c,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1994-02-17 08:57:38 PST
Newsgroups: comp.sources.atari.st
Path: gmd.de!newsserver.jvnc.net!netnews.upenn.edu!dsinc!spool.mu.edu!
howland.reston.ans.net!EU.net!sun4nl!cwi.nl!avg
From: tele@tubul.limmat.net.ch (Patrick Seemann)
Subject: v07i006: tos3.06swg.tbl -- Keyboard Table
Message-ID: <csas.7.6@cwi.nl>
Originator: avg@mijt.cwi.nl
Sender: news@cwi.nl (The Daily Dross)
Nntp-Posting-Host: mijt.cwi.nl
Organization: CWI, Amsterdam
Date: Thu, 17 Feb 1994 16:57:38 GMT
Approved: avg@cwi.nl
Lines: 155

Submitted-by: tele@tubul.limmat.net.ch (Patrick Seemann)
Posting-number: Volume 7, Issue 6
Archive-name: tables/tos3.06swg.tbl


---------------------------------------------------------------------

I wouldn't call the SWG (swiss german) TOS exotic but some things are really
a bit strange (see below). A lot of work is done in the keyboard driver, to
include a number of special characters like german/french umlaute, pound
sign etc.

gruss, pat

--
/----------------------+-----------------------------------------\
| Patrick Seemann      | tele@tubul.limmat.net.ch                |
| Zypressenstrasse 67  +-----------------------------------------+
| CH-8004 Zurich       | -*-*- "One planet is all you get!" -*-*- |
```

```
\----------------------+-----------------------------------------/

--------------------------------------------------------------------------
TOS 3.06 for Switzerland (German)
     unshift       shift         capslock
  0 =    0x00     0x00     0x00
  1 =    0x1b     0x1b     0x1b
  2 =    0x2b "+" 0x31 "1" 0x31 "1"
  3 =    0x22 """ 0x32 "2" 0x32 "2"
  4 =    0x2a "*" 0x33 "3" 0x33 "3"
  5 =    0x87     0x34 "4" 0x34 "4"
  6 =    0x25 "%" 0x35 "5" 0x35 "5"
  7 =    0x26 "&" 0x36 "6" 0x36 "6"
  8 =    0x2f "/" 0x37 "7" 0x37 "7"
  9 =    0x28 "(" 0x38 "8" 0x38 "8"
 10 =    0x29 ")" 0x39 "9" 0x39 "9"
 11 =    0x3d "=" 0x30 "0" 0x30 "0"
 12 =    0x3f "?" 0x27 "'" 0x27 "'"
 13 =    0x60 "`" 0x5e "^" 0x5e "^"
 14 =    0x08     0x08     0x08
 15 =    0x09     0x09     0x09
 16 =    0x51 "Q" 0x71 "q" 0x51 "Q"
 17 =    0x57 "W" 0x77 "w" 0x57 "W"
 18 =    0x45 "E" 0x65 "e" 0x45 "E"
 19 =    0x52 "R" 0x72 "r" 0x52 "R"
 20 =    0x54 "T" 0x74 "t" 0x54 "T"
 21 =    0x5a "Z" 0x7a "z" 0x5a "Z"
 22 =    0x55 "U" 0x75 "u" 0x55 "U"
 23 =    0x49 "I" 0x69 "i" 0x49 "I"
 24 =    0x4f "O" 0x6f "o" 0x4f "O"
 25 =    0x50 "P" 0x70 "p" 0x50 "P"
 26 =    0x8a     0x81     0x9a
 27 =    0x21 "!" 0xb9     0xb9
 28 =    0x0d     0x0d     0x0d
 29 =    0x00     0x00     0x00
 30 =    0x41 "A" 0x61 "a" 0x41 "A"
 31 =    0x53 "S" 0x73 "s" 0x53 "S"
 32 =    0x44 "D" 0x64 "d" 0x44 "D"
 33 =    0x46 "F" 0x66 "f" 0x46 "F"
 34 =    0x47 "G" 0x67 "g" 0x47 "G"
 35 =    0x48 "H" 0x68 "h" 0x48 "H"
 36 =    0x4a "J" 0x6a "j" 0x4a "J"
 37 =    0x4b "K" 0x6b "k" 0x4b "K"
 38 =    0x4c "L" 0x6c "l" 0x4c "L"
 39 =    0x82     0x94     0x99
 40 =    0x85     0x84     0x8e
 41 =    0xf8     0xdd     0xdd
 42 =    0x00     0x00     0x00
 43 =    0x9c     0x24 "$" 0x24 "$"
 44 =    0x59 "Y" 0x79 "y" 0x59 "Y"
 45 =    0x58 "X" 0x78 "x" 0x58 "X"
 46 =    0x43 "C" 0x63 "c" 0x43 "C"
 47 =    0x56 "V" 0x76 "v" 0x56 "V"
```

```
 48 =    0x42 "B" 0x62 "b" 0x42 "B"
 49 =    0x4e "N" 0x6e "n" 0x4e "N"
 50 =    0x4d "M" 0x6d "m" 0x4d "M"
 51 =    0x3b ";" 0x2c "," 0x2c ","
 52 =    0x3a ":" 0x2e "." 0x2e "."
 53 =    0x5f "_" 0x2d "-" 0x2d "-"
 54 =    0x00     0x00     0x00
 55 =    0x00     0x00     0x00
 56 =    0x00     0x00     0x00
 57 =    0x20 " " 0x20 " " 0x20 " "
 58 =    0x00     0x00     0x00
 59 =    0x00     0x00     0x00
 60 =    0x00     0x00     0x00
 61 =    0x00     0x00     0x00
 62 =    0x00     0x00     0x00
 63 =    0x00     0x00     0x00
 64 =    0x00     0x00     0x00
 65 =    0x00     0x00     0x00
 66 =    0x00     0x00     0x00
 67 =    0x00     0x00     0x00
 68 =    0x00     0x00     0x00
 69 =    0x00     0x00     0x00
 70 =    0x00     0x00     0x00
 71 =    0x37 "7" 0x00     0x00
 72 =    0x38 "8" 0x00     0x00
 73 =    0x00     0x00     0x00
 74 =    0x2d "-" 0x2d "-" 0x2d "-"
 75 =    0x34 "4" 0x00     0x00
 76 =    0x00     0x00     0x00
 77 =    0x36 "6" 0x00     0x00
 78 =    0x2b "+" 0x2b "+" 0x2b "+"
 79 =    0x00     0x00     0x00
 80 =    0x32 "2" 0x00     0x00
 81 =    0x00     0x00     0x00
 82 =    0x30 "0" 0x00     0x00
 83 =    0x7f     0x7f     0x7f
 84 =    0x00     0x00     0x00
 85 =    0x00     0x00     0x00
 86 =    0x00     0x00     0x00
 87 =    0x00     0x00     0x00
 88 =    0x00     0x00     0x00
 89 =    0x00     0x00     0x00
 90 =    0x00     0x00     0x00
 91 =    0x00     0x00     0x00
 92 =    0x00     0x00     0x00
 93 =    0x00     0x00     0x00
 94 =    0x00     0x00     0x00
 95 =    0x00     0x00     0x00
 96 =    0x3e ">" 0x3c "<" 0x3c "<"
 97 =    0x00     0x00     0x00
 98 =    0x00     0x00     0x00
 99 =    0x28 "(" 0x28 "(" 0x28 "("
100 =    0x29 ")" 0x29 ")" 0x29 ")"
```

```
101 =    0x2f "/" 0x2f "/" 0x2f "/"
102 =    0x2a "*" 0x2a "*" 0x2a "*"
103 =    0x37 "7" 0x37 "7" 0x37 "7"
104 =    0x38 "8" 0x38 "8" 0x38 "8"
105 =    0x39 "9" 0x39 "9" 0x39 "9"
106 =    0x34 "4" 0x34 "4" 0x34 "4"
107 =    0x35 "5" 0x35 "5" 0x35 "5"
108 =    0x36 "6" 0x36 "6" 0x36 "6"
109 =    0x31 "1" 0x31 "1" 0x31 "1"
110 =    0x32 "2" 0x32 "2" 0x32 "2"
111 =    0x33 "3" 0x33 "3" 0x33 "3"
112 =    0x30 "0" 0x30 "0" 0x30 "0"
113 =    0x2e "." 0x2e "." 0x2e "."
114 =    0x0d     0x0d     0x0d
115 =    0x00     0x00     0x00
116 =    0x00     0x00     0x00
117 =    0x00     0x00     0x00
118 =    0x00     0x00     0x00
119 =    0x00     0x00     0x00
120 =    0x00     0x00     0x00
121 =    0x00     0x00     0x00
122 =    0x00     0x00     0x00
123 =    0x00     0x00     0x00
124 =    0x00     0x00     0x00
125 =    0x00     0x00     0x00
126 =    0x00     0x00     0x00
127 =    0x00     0x00     0x00
------------------------------------------------------------------------
```

From 6640174496983314924
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,91de3f0df692c37a,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,91de3f0df692c37a,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1994-05-10 05:19:24 PST
Newsgroups: comp.sources.atari.st,comp.sys.atari.st
Path: bga.com!news.sprintlink.net!rtp.vnet.net!mary.iia.org!babbage.ece.uc.edu!
news.kei.com!MathWorks.Com!europa.eng.gtefsd.com!howland.reston.ans.net!EU.net!
sun4nl!cwi.nl!avg
From: avg@cwi.nl (Annius Groenink)
Subject: v08INF1: Introduction to comp.sources.atari.st
Message-ID: <CpL44o.A2M@cwi.nl>
Originator: avg@mijt.cwi.nl
Sender: news@cwi.nl (The Daily Dross)
Nntp-Posting-Host: mijt.cwi.nl
Organization: CWI, Amsterdam
Date: Tue, 10 May 1994 11:34:00 GMT
Approved: avg@cwi.nl (Annius Groenink)
Expires: Wed, 7 Sep 1994 11:34:00 GMT

Lines: 162
Xref: bga.com comp.sources.atari.st:12 comp.sys.atari.st:10242

Submitted-by: avg@cwi.nl (Annius Groenink)
Posting-number: Volume 8, Info 1

This article is posted regularly. It gives a short introduction to comp.sources.atari.st and describes how to submit sources or digressions to the group.

I am always looking for suggestions on how to improve the usefulness of the newsgroup, and can be contacted at the following internet address:

avg@cwi.nl

Annius Groenink, Moderator.

ABOUT comp.sources.atari.st

The moderated usenet group comp.sources.atari.st contains publications or announcements of sources and texts about programming for Atari TOS computers. The publications on comp.sources.atari.st may be freely distributed as long as modifications are documented and the original authors are mentioned.

THIS IS A MODERATED GROUP. YOU CANNOT POST TO IT BY YOURSELF. Read the instructions below for submitting material intended for publication on comp.sources.atari.st.

SUBMITTING SOURCES OR DIGRESSIONS FOR PUBLICATION

Items intended for posting or queries and problem notes should be sent to avg@cwi.nl. If you are on a UUCP-only site, you can send them to uunet!cwi.nl!avg.

If you want verification of arrival, say so at the beginning of your submission, or in a cover note. I try to verify that a program works, and if I can't get it to work, I may hold up posting it for a couple of days. Please note that, except in rare cases, software without documentation will not be published. The backlog from receipt to posting varies from one to four weeks depending mostly on the set of submissions currently in my queue.

If you are submitting both sources and binaries, PLEASE send the two separately. If I have to separate your sources from your binaries by hand, your submission will most likely sit on the back burner for a while, or be rejected outright.

Sources can be submitted in ASCII form, but for large pacages, a ZOO 2.1 archive encoded using Dumas uuencode is preferred.

Please include a description of your program at the top of your article. Look at old issues of comp.binaries.atari.st for examples of the sorts of descriptions I'm looking for. In general you should indicate what the program does, on what types of Atari TOS computers, whether your program requires a certain amount of on-board memory, a hard disk or a certain screen resolution. If necessary, I may rewrite or add to this information in order to make the postings to the group more uniform. I can't check your submission in all possible configurations, and see what works and what doesn't. Without a description, your submission is likely to be rejected.

THE STRUCTURE OF COMP.SOURCES.ATARI.ST ARTICLES

There are three types of articles in comp.binaries.atari.st: "information postings", "sources" and "announcements". The source postings are also called "Issues". The type of a posting can be distinguished by the subject line:

```
Subject: v02INF1: Introduction to comp.sources.atari.st
```

This first word in the title identifies this as the first info posting of volume two. Similarly, the subject line shown below:

```
Subject: v06i081: godsrc -- Universe simulator
```

identifies this as the 81st source article in Volume 6. Large sources are broken up into smaller pieces, and have subject lines that look like this:

```
Subject: v06i088: decbustr -- VMS clone in ST-LOGO, Part03/08
```

Announcements of large source packages uploaded to ftp archives are also considered for publication. The subject header for such files looks like

```
Subject: v07a007: GCC 7.9.4 -- Same as 7.9.3, but considerably slower
```

Certain information about the system configuration required to use the program is given on the keywords line.

```
Keywords: MWC, Megamax, 1meg, high
```

This means that the program compiles with Mark Williams or Megamax C, requires at least one meg of RAM and only runs in high resolution. Following is a list of keywords; new ones may be added as needed. They are mostly self- explanatory.

```
  Alcyon  - Alcyon C (included with Atari developer's kit)
  MWC         - Mark Williams C
  Megamax - Megamax C
  Laser       - Laser C
  Lattice - Lattice C
  C       - Any C compiler
  OSS         - OSS Personal Pascal
  BASIC       - ST BASIC (this is not a joke!)
  uuencode    - program is uuencoded (uudecode required to unpack)
  arc         - program is archived (arc required to unpack)
  zoo-high    - program is archived (Zoo 2.1 required to unpack)
  lharc       - program is archived (lharc required to unpack)
  high        - high resolution
  medium  - medium resolution
  low         - low resolution
  1meg        - needs 1 meg of RAM
```

The References: line, if present, points to the previous part of a multi-part program, so that newsreaders (trn, nn, etc.) which can follow threads will recognize a large program as a single thread.

The first few lines of an article are auxiliary headers that look like this:

```
  Submitted-by: david@bdt.UUCP (David Beckmeyer)
  Posting-number: Volume 6, Issue 80
  Archive-name: micro-rtx
```

The "Submitted by" is the author of the program. If you have comments about the sources published in comp.sources.atari.st this is the person to contact. When possible, this address is in domain form, otherwise it is a UUCP bang path relative to some major (backbone) site.

The second line repeats the volume/issue information for the aide of notes sites and automatic archiving programs.

The Archive-name is the "official" name of this source in the archive. Large postings will have names that look like this:

```
Archive-name: desktop/part01
```

REPORTING AND TRACKING BUGS AND PATCHES TO POSTINGS

Most of the time, bugs and updates will be posted to comp.sys.atari.st. Sometimes, "official" updates from the author will be posted submitted to comp.sources.atari.st so that they are archived with the sources.

Since there is no easy way to apply source code patches on the ST, changes to a program often result in a whole new version being posted.

To report bugs, contact the person listed in the Submitted-to header. Often there is a contact address in a README file, too. I do not maintain the sources I moderate, so don't send your bug reports to me.

From -5651653285450903945
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,8264b1014e664586,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,8264b1014e664586,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1994-08-15 05:57:40 PST
Newsgroups: comp.sources.atari.st,comp.sys.atari.st
Path: bga.com!news.sprintlink.net!uunet!EU.net!sun4nl!cwi.nl!avg
From: avg@cwi.nl (Annius Groenink)
Subject: v08INF2: Introduction to comp.sources.atari.st
Message-ID: <CuKtKn.FKs@cwi.nl>
Originator: avg@mijt.cwi.nl
Sender: news@cwi.nl (The Daily Dross)
Nntp-Posting-Host: mijt.cwi.nl
Organization: CWI, Amsterdam
Date: Mon, 15 Aug 1994 12:33:59 GMT
Approved: avg@cwi.nl (Annius Groenink)
Expires: Tue, 13 Dec 1994 12:33:59 GMT
Lines: 169
Xref: bga.com comp.sources.atari.st:15 comp.sys.atari.st:15539

Submitted-by: avg@cwi.nl (Annius Groenink)
Posting-number: Volume 8, Info 2

This article is posted regularly. It gives a short introduction to comp.sources.atari.st and describes how to submit sources or digressions to the group.

I am always looking for suggestions on how to improve the usefulness of the newsgroup, and can be contacted at the following internet address:

avg@cwi.nl

Annius Groenink, Moderator.

ABOUT comp.sources.atari.st

The moderated usenet group comp.sources.atari.st contains publications or announcements of sources and texts about programming for Atari TOS computers. The publications on comp.sources.atari.st may be freely

distributed as long as modifications are documented and the original authors are mentioned.

THIS IS A MODERATED GROUP. YOU CANNOT POST TO IT BY YOURSELF. Read the instructions below for submitting material intended for publication on comp.sources.atari.st.

SUBMITTING SOURCES OR DIGRESSIONS FOR PUBLICATION

Items intended for posting or queries and problem notes should be sent to avg@cwi.nl. If you are on a UUCP-only site, you can send them to uunet!cwi.nl!avg.

If you want verification of arrival, say so at the beginning of your submission, or in a cover note. I try to verify that a program works, and if I can't get it to work, I may hold up posting it for a couple of days. Please note that, except in rare cases, software without documentation will not be published. The backlog from receipt to posting varies from one to four weeks depending mostly on the set of submissions currently in my queue.

If you are submitting both sources and binaries, PLEASE send the two separately. If I have to separate your sources from your binaries by hand, your submission will most likely sit on the back burner for a while, or be rejected outright.

Sources can be submitted in ASCII form, but for large pacages, a ZOO 2.1 archive encoded using Dumas uuencode is preferred.

Please include a description of your program at the top of your article. Look at old issues of comp.binaries.atari.st for examples of the sorts of descriptions I'm looking for. In general you should indicate what the program does, on what types of Atari TOS computers, whether your program requires a certain amount of on-board memory, a hard disk or a certain screen resolution. If necessary, I may rewrite or add to this information in order to make the postings to the group more uniform. I can't check your submission in all possible configurations, and see what works and what doesn't. Without a description, your submission is likely to be rejected.

THE STRUCTURE OF COMP.SOURCES.ATARI.ST ARTICLES

There are two types of articles in comp.binaries.atari.st: "information postings" and "sources". The source postings are also called "Issues". For a while, 'announcements' were also accepted, but there is now a dedicated group, comp.sys.atari.announce, for that purpose. The type of a posting can be distinguished by the subject line:

```
  Subject: v02INF1: Introduction to comp.sources.atari.st
```

This first word in the title identifies this as the first info posting of volume two. Similarly, the subject line shown below:

```
  Subject: v06i081: godsrc -- Universe simulator
```

identifies this as the 81st source article in Volume 6. Large sources are broken up into smaller pieces, and have subject lines that look like this:

```
  Subject: v06i088: decbustr -- VMS clone in ST-LOGO, Part03/08
```

Certain information about the system configuration required to use the program is given on the keywords line.

```
  Keywords: MWC, Megamax, 1meg, high
```

This means that the program compiles with Mark Williams or Megamax C, requires at least one meg of RAM and only runs in high resolution. Following is a list of keywords; new ones may be added as needed. They are mostly self- explanatory.

```
  Alcyon  - Alcyon C (included with Atari developer's kit)
  MWC         - Mark Williams C
  Megamax - Megamax C
  Laser       - Laser C
  Lattice - Lattice C
  C       - Any C compiler
  OSS         - OSS Personal Pascal
  BASIC       - ST BASIC (this is not a joke!)
  uuencode    - program is uuencoded (uudecode required to unpack)
  arc         - program is archived (arc required to unpack)
  zoo-high    - program is archived (Zoo 2.1 required to unpack)
  lharc       - program is archived (lharc required to unpack)
  high        - high resolution
  medium  - medium resolution
  low         - low resolution
  1meg        - needs 1 meg of RAM
```

The References: line, if present, points to the previous part of a multi-part program, so that newsreaders (trn, nn, etc.) which can follow threads will recognize a large program as a single thread.

The first few lines of an article are auxiliary headers that look like this:

```
  Submitted-by: david@bdt.UUCP (David Beckmeyer)
  Posting-number: Volume 6, Issue 80
  Archive-name: micro-rtx
```

The "Submitted by" is the author of the program. If you have comments about the sources published in comp.sources.atari.st this is the person to contact. When possible, this address is in domain form, otherwise it is a UUCP bang path relative to some major (backbone) site.

The second line repeats the volume/issue information for the aide of notes sites and automatic archiving programs.

The Archive-name is the "official" name of this source in the archive. Large postings will have names that look like this:

Archive-name: desktop/part01

REPORTING AND TRACKING BUGS AND PATCHES TO POSTINGS

Most of the time, bugs and updates will be posted to comp.sys.atari.st. Sometimes, "official" updates from the author will be posted submitted to comp.sources.atari.st so that they are archived with the sources.

Since there is no easy way to apply source code patches on the ST, changes to a program often result in a whole new version being posted.

To report bugs, contact the person listed in the Submitted-to header. Often there is a contact address in a README file, too. I do not maintain the sources I moderate, so don't send your bug reports to me.

--
Annius V. Groenink | avg@cwi.nl | Private/Edith/ZFC:
CWI, Kruislaan 413 | A.V.Groenink@zfc.nl | P.O. Box 12079
1098 SJ Amsterdam | Room M233 ext. 4077 | NL 1100 AB Amsterdam
The Netherlands | Phone: +31 20 592 4077 | Phone: +31 20 695 9901
--
Annius V. Groenink | avg@cwi.nl | Private/Edith/ZFC:
CWI, Kruislaan 413 | A.V.Groenink@zfc.nl | P.O. Box 12079
1098 SJ Amsterdam | Room M233 ext. 4077 | NL 1100 AB Amsterdam
The Netherlands | Phone: +31 20 592 4077 | Phone: +31 20 695 9901

From -5330801830298053732
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,29de92bed5098f73,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,29de92bed5098f73,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1994-09-19 21:03:06 PST
Newsgroups: comp.sources.atari.st,comp.sys.atari.st
Path: bga.com!news.sprintlink.net!howland.reston.ans.net!EU.net!sun4nl!cwi.nl!avg
From: avg@cwi.nl (Annius Groenink)
Subject: v08INF2: Introduction to comp.sources.atari.st

Message-ID: <CwDGKv.BI1@cwi.nl>
Originator: avg@mijt.cwi.nl
Sender: news@cwi.nl (The Daily Dross)
Nntp-Posting-Host: mijt.cwi.nl
Organization: CWI, Amsterdam
Date: Mon, 19 Sep 1994 10:17:18 GMT
Approved: avg@cwi.nl (Annius Groenink)
Expires: Tue, 17 Jan 1995 10:17:18 GMT
Lines: 164
Xref: bga.com comp.sources.atari.st:16 comp.sys.atari.st:17268

Submitted-by: avg@cwi.nl (Annius Groenink)
Posting-number: Volume 8, Info 2

This article is posted regularly. It gives a short introduction to comp.sources.atari.st and describes how to submit sources or digressions to the group.

I am always looking for suggestions on how to improve the usefulness of the newsgroup, and can be contacted at the following internet address:

avg@cwi.nl

Annius Groenink, Moderator.

ABOUT comp.sources.atari.st

The moderated usenet group comp.sources.atari.st contains publications or announcements of sources and texts about programming for Atari TOS computers. The publications on comp.sources.atari.st may be freely distributed as long as modifications are documented and the original authors are mentioned.

THIS IS A MODERATED GROUP. YOU CANNOT POST TO IT BY YOURSELF. Read the instructions below for submitting material intended for publication on comp.sources.atari.st.

SUBMITTING SOURCES OR DIGRESSIONS FOR PUBLICATION

Items intended for posting or queries and problem notes should be sent to avg@cwi.nl. If you are on a UUCP-only site, you can send them to uunet!cwi.nl!avg.

If you want verification of arrival, say so at the beginning of your submission, or in a cover note. I try to verify that a program works, and if I can't get it to work, I may hold up posting it for a couple of days. Please note that, except in rare cases, software without

documentation will not be published. The backlog from receipt to posting varies from one to four weeks depending mostly on the set of submissions currently in my queue.

If you are submitting both sources and binaries, PLEASE send the two separately. If I have to separate your sources from your binaries by hand, your submission will most likely sit on the back burner for a while, or be rejected outright.

Sources can be submitted in ASCII form, but for large pacages, a ZOO 2.1 archive encoded using Dumas uuencode is preferred.

Please include a description of your program at the top of your article. Look at old issues of comp.binaries.atari.st for examples of the sorts of descriptions I'm looking for. In general you should indicate what the program does, on what types of Atari TOS computers, whether your program requires a certain amount of on-board memory, a hard disk or a certain screen resolution. If necessary, I may rewrite or add to this information in order to make the postings to the group more uniform. I can't check your submission in all possible configurations, and see what works and what doesn't. Without a description, your submission is likely to be rejected.

THE STRUCTURE OF COMP.SOURCES.ATARI.ST ARTICLES

There are two types of articles in comp.binaries.atari.st: "information postings" and "sources". The source postings are also called "Issues". For a while, 'announcements' were also accepted, but there is now a dedicated group, comp.sys.atari.announce, for that purpose. The type of a posting can be distinguished by the subject line:

```
Subject: v02INF1: Introduction to comp.sources.atari.st
```

This first word in the title identifies this as the first info posting of volume two. Similarly, the subject line shown below:

```
Subject: v06i081: godsrc -- Universe simulator
```

identifies this as the 81st source article in Volume 6. Large sources are broken up into smaller pieces, and have subject lines that look like this:

```
Subject: v06i088: decbustr -- VMS clone in ST-LOGO, Part03/08
```

Certain information about the system configuration required to use the program is given on the keywords line.

```
Keywords: MWC, Megamax, 1meg, high
```

This means that the program compiles with Mark Williams or Megamax C, requires at least one meg of RAM and only runs in high resolution. Following is a list of keywords; new ones may be added as needed. They are mostly self- explanatory.

```
  Alcyon  - Alcyon C (included with Atari developer's kit)
  MWC         - Mark Williams C
  Megamax - Megamax C
  Laser       - Laser C
  Lattice - Lattice C
  C       - Any C compiler
  OSS         - OSS Personal Pascal
  BASIC       - ST BASIC (this is not a joke!)
  uuencode    - program is uuencoded (uudecode required to unpack)
  arc         - program is archived (arc required to unpack)
  zoo-high    - program is archived (Zoo 2.1 required to unpack)
  lharc       - program is archived (lharc required to unpack)
  high        - high resolution
  medium  - medium resolution
  low         - low resolution
  1meg        - needs 1 meg of RAM
```

The References: line, if present, points to the previous part of a multi-part program, so that newsreaders (trn, nn, etc.) which can follow threads will recognize a large program as a single thread.

The first few lines of an article are auxiliary headers that look like this:

```
  Submitted-by: david@bdt.UUCP (David Beckmeyer)
  Posting-number: Volume 6, Issue 80
  Archive-name: micro-rtx
```

The "Submitted by" is the author of the program. If you have comments about the sources published in comp.sources.atari.st this is the person to contact. When possible, this address is in domain form, otherwise it is a UUCP bang path relative to some major (backbone) site.

The second line repeats the volume/issue information for the aide of notes sites and automatic archiving programs.

The Archive-name is the "official" name of this source in the archive. Large postings will have names that look like this:

```
Archive-name: desktop/part01
```

REPORTING AND TRACKING BUGS AND PATCHES TO POSTINGS

Most of the time, bugs and updates will be posted to comp.sys.atari.st.

Sometimes, "official" updates from the author will be posted submitted to comp.sources.atari.st so that they are archived with the sources.

Since there is no easy way to apply source code patches on the ST, changes to a program often result in a whole new version being posted.

To report bugs, contact the person listed in the Submitted-to header. Often there is a contact address in a README file, too. I do not maintain the sources I moderate, so don't send your bug reports to me.

--
Annius V. Groenink | avg@cwi.nl | Private/Edith/ZFC:
CWI, Kruislaan 413 | A.V.Groenink@zfc.nl | P.O. Box 12079
1098 SJ Amsterdam | Room M233 ext. 4077 | NL 1100 AB Amsterdam
The Netherlands | Phone: +31 20 592 4077 | Phone: +31 20 695 9901

From -5850481309230008784
X-Google-Language: ENGLISH,ASCII
X-Google-Thread: 100493,f63aceb9bb33819b,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1994-09-19 21:03:18 PST
Newsgroups: comp.sources.atari.st
Path: bga.com!news.sprintlink.net!howland.reston.ans.net!EU.net!sun4nl!cwi.nl!avg
From: hopper_t@msupa.pa.msu.edu
Subject: v08i001: desktips -- detailed description of NEWDESK and DESKTOP.INF files part 01/01
Message-ID: <CwDGL5.BIt@cwi.nl>
Originator: avg@mijt.cwi.nl
Keywords: English
Sender: news@cwi.nl (The Daily Dross)
Nntp-Posting-Host: mijt.cwi.nl
Organization: CWI, Amsterdam
Date: Mon, 19 Sep 1994 10:17:28 GMT
Approved: avg@cwi.nl (Annius Groenink)
Lines: 869

Submitted-by: hopper_t@msupa.pa.msu.edu
Posting-number: Volume 8, Issue 1
Archive-name: desktips/part01

ATARI TOS DESKTOP SURVIVAL KIT

****

A GUIDE TO THE DESKTOP.INF AND NEWDESK.INF FILES

by Thomas J Hopper

CONTENTS

Legalities	-	Rights, etc.
Welcome	-	An introduction of sorts
DESKTOP/NEWDESK Basics	-	Get to know the INF files
Editing the INF File	-	Basics about editing the INF files
The Keyboard and Installing Apps	-	Keyboard functions
Neat Hacks	-	Some suggestions
INF File Composition	-	Explanation of INF file structure
Annotated NEWDESK.INF file	-	A real-life example

LEGALITIES

I make no warranty as to the usefulness or accuracy of this document.

copyright © 1994 Thomas J Hopper
This file may only be distributed in its entirety and at no charge.
No portion of this document may be distributed for profit without the
explicit permission of the author.

WELCOME!

There are lots of things you can do with the new Atari desktops (and with the old ones!); almost as much as with a replacement desktop like NeoDesk© from Gribnif Software! Unfortunately, the manuals Atari gives us with the computers don't tell you about all of these great features. I will try to rectify this deficiency.

I have compiled as much information on the NEWDESK.INF file that I could, and thrown in some info on TOS 2.06. This file focuses on Atari's NEWDESK.INF file, and in particular on TOS 2.06, since they are what I have. If you don't have TOS 2.06, don't worry; most of the really useful info in here is applicable to *all* versions of TOS, and some info specific to TOS 4.0x is included, too. I have tried to point out where information may be specific only to certain versions of TOS. With this as a reference, you should be able to modify your DESKTOP/NEWDESK.INF file to do almost anything with your desktop!

I have learned all of this partly through trial and error, and partly through discussions on online services. A lot of the information came to me by way of the NEWDESK topic on GEnie in the Atari Roundtable. GEnie, by the way, is an excellent source of information and support for the Atari line of computers; being the only official online Atari service, there are more Atarians there than anywhere else. I highly recommend this service. I suppose I should state that I have no connection with GEnie other than as a satisfied user. Another excellent source of info and programs is the Internet, where Atarians from all over the world can (and do!) converse and share programs.

If you find that something is wrong, please let me know! If you see something I missed, or if you think I should include some information that isn't in here, let me know that, too. And of course, suggestions are always welcome. When contacting me, include your machine type, TOS, GEM, AES, MiNT, and MultiTOS versions (whichever apply). This will help me decipher different, unknown parameters. If you don't know some of this information or don't have MiNT or something, don't let that stop you from contacting me! But please, try to include at least the TOS version (TOS date will suffice, if necessary; just give me the copyright dates in the "Desktop Info..." menu). Also include your name and some way that I can get in touch with you, such as email address or postal address.

I can be reached at:

GEnie T.HOPPER
Internet hopper_t@msupa.pa.msu.edu
or
Thomas Hopper
2910 BeauJardin #203
Lansing, MI, USA 48910

DESKTOP/NEWDESK BASICS

The DESKTOP.INF and NEWDESK.INF files in Atari computers contain all the information your computer needs to configure itself. It contains all the names, positions and (for appropriate TOS versions) icon numbers for your desktop icons. It stores all the info on Installed Apps, desktop icons, and window positions (even for windows that have been closed!). If you leave a window open when saving the desktop, the location, current directory, and file mask (for those versions of TOS that support masks) are saved. It stores info on whether or not the key click and system bell are turned on, the repeat rate for key clicks, mouse double click rate and tracking rate, current resolution, keyboard shortcuts for menu items (in TOS >= 2.0x), and lots more. In short, it stores every peice of information that you can configure from the desktop or the standard Atari Control Panel.

When you boot up your machine, the computer loads in auto programs and then reads the DESKTOP/NEWDESK.INF (hereafter referred to collectively

as simply "INF") file for information on resolution, windows, etc. If you have Atari's Control Panel (or a replacement), the key repeat rate, printer and serial port configuration, and other information is also loaded from the INF file and configured.

To change the INF file, you need simply alter a setting from either the Control Panel (not the extensible control panel, however) or one of the desktop menus, and then select "Save Desktop." If you do this several times and display the INF file after each save, you will get a feel for which changes to the desktop correspond to changes in the INF file. Before doing this, of course, make a backup of your original INF file by renaming it to DESKTOP.INX. This way if you somehow "break" the INF file, you can always be sure to have a working one handy.

You can view the file directly from the desktop; the INF file is simply an ASCII file! This means that not only is everything in the file plainly visible from the desktop, but you can load it into just about any word processor, text editor, or DTP program and edit it by hand. This has the advantage of allowing you to make certain changes that the desktop doesn't allow for. If you're really familiar with the INF file, you can also make multiple changes in much less time than it would take you from the desktop. The editor, however, must be able to load and save files in ASCII format, and it must not strip trailing spaces from lines (more on this in a moment).

One important bit of information to know and understand when editing the INF file is how TOS reads and applies the information in it. Not surprisingly, the INF file acts as a filter, through which TOS sifts programs and files to apply various actions (running files, assigning icons, etc.). What *is* surprising is that these filters work in reverse order from what you see in the INF file. The *last* entry in the INF file is the *first* filter applied! The way this works is as follows:

> For icon assignments, TOS reads from the end of the file to the beginning, comparing each file in a directory to each filter (icon assignment) until a match is found. When TOS finds a match, it moves on to the next file. So if you have a program SQUISHII.APP, and assign an icon to it, that assignment should go toward the end of the file, that way it will be filtered out early on. If you have an icon assingment like #I xx xx xxx @ *.*@ @ *below* the assignment for SQUISHII.APP, then SQUISHII.APP will be filtered out at the *.* (and assigned that icon) rather than at #I xx xx xxx @ SQUISHII.APP@ @ . Hence, you would never see the special icon you assigned to SQUISHII.APP. This is important to understand, for two reasons. The first is that when editing the INF file, you want to work down the file from most general to most specific; getting this order wrong will give you results other than what you wanted. The other reason is that when you make an icon assignment from the desktop, it is saved at the end of the INF file. If you were to make lots of assignments, then do a generic (*.*) icon assignment, all your files would use the generic icon (until you edited the INF

file to have *.* be the last filter applied).

For program assignments, TOS reads the INF file in the same way as for icons, and applies filters in the same way. When you double click on a file (whether it's executable or not), TOS starts looking for matches from the bottom of the INF file and works its way to the top. If you were to install a program VIEWER.APP for file type *.*, this assignment would be saved at the end to the INF file, and hence be the first filter TOS saw; all files, no matter what type they were, would cause VIEWER.APP to run and they would be loaded in to it. This is true even for PRGs, APPs, TOSs, etc.; in the INF file, these are just treated as applications installed with no file type. As with icons, if you edit the INF file make sure you work from most general at the top of the INF file to most specific at the bottom.

EDITING THE INF FILE

Before you begin editing your INF file, make sure that you have a backup of you current, working version! It is possible (and fairly easy, really) to change the INF file so that the computer will become "confused" and not boot!

Probably the first thing to remember when editing your INF file is that NEWDESK.INF can't be larger than 4kb (4096 bytes), and starts causing trouble with the system around 3900 bytes. Similarly, the DESKICON.RSC can only be 64kb (65536 bytes). The newer DESKCICN.RSC (the color icon resource for TOS 4.0x and up) can be larger than this, though I don't know if there is a limit. If you try to exceed the limit, you'll get "Out Of Memory" errors and lots of headaches.

On earlier versions of TOS (less than 2.0x), the limit on DESKTOP.INF is even more restrictive; it can't be larger than 1kb (1024 bytes)! However, you also don't have a lot of icon assignments eating up that space, so it's a fair trade.

One excellent way around this is to have a directory full of different INF files; a specific one for each task. If you put only those application and icon assignments necessary for a particular task, you should have no problem avoiding the size limit. To switch between INF files, then you can just install either Charles F Johnson's shareware product Desk Switch 1.1 or Klaus Pederson's public domain Load Inf as the application for file types of *.INF. That way, when you want to change to a different task, just double click on the appropriate INF file, and away you go with all the key assignments, installed apps, desktop icons, and icon assignments you want for that task.

Another way around this is to get a program like GEMRAM, which loads GEM into RAM, and the program Shell Buffer (SHBUFxxx.PRG), which lets you configure the allowed size of the INF file.

Both solutions should work great on any Atari, and both have their advantages and disadvantages. Give them both a try and see which works best for you.

Something else to keep in mind when editing your INF file is that TOS expects to find certain formats at certain locations in the file. Line #d, for instance, has to have a certain number of spaces in it since TOS looks roughly 124 bytes into the INF file for the next line (#Z if you have TOS >=1.04 and a program set to auto boot, or #K for the menu key equivalents). That number for the bytes, incidently, also includes carriage returns (EOL characters). Before changing your desktop file, be sure you know *exactly* how many spaces belong in line #d; the wrong number can produce unpredictable results and fatal errors. To find out, of course, just load the file into a text editor that doesn't strip trailing spaces (Word Writer, Alice, Everest, and tons more) and start counting!

Most other lines (but not all of them) in the INF file require a trailing space, so if you edit your file be sure to know which ones need a space and which ones don't.

What this all means, of course, is that if you edit your DESKTOP.INF or NEWDESK.INF file make sure you're using an editor that doesn't strip trailing spaces, and which saves files as ASCII text. And, as always, KEEP A BACKUP OF YOUR ORIGINAL INF FILE!

THE KEYBOARD AND INSTALLING APPS

With the newer TOS versions, you can open a drive into a window by pressing <Alt> and the drive letter. If you want to open a drive into the current top window, you don't have to close the window and then open the drive; just hold down <Cntrl> and press the drive letter! The window will automatically be changed to the root directory of whatever drive you selected.

With TOS versions 2.0x and up, Atari has made it possible to assign a keyboard equivalent to any of the desktop menu items. That way you can select a file and hit "I" to get info on it. Unfortunately, the mneumonics can get pretty complicated, since there are also items that could use "I" like "Show as Icons" and "Install Icon." "D" could be used for "Sort by Date," "Delete," and "Install Devices." The desktop appears to only allow normal characters (capital A through Z) for these assignments, which gets pretty limiting. Luckly, you can also use control-key combinations! When changing the menu assignments in the Desktop Configuration menu, just hold down the <control> key while pressing your key. Now, instead of accidently deleting a file by pressing "D," you can assign <cntrl><delete> (or <cntrl><D>, if yo prefer) to "Delete," and never have to worry about accidentally deleting a file!

Take note! If you use a control-key combo for a particular menu item,

that combo will no longer be available for normal desktop usage. For instance, if you assign <cntrl><D> to "Show by Date," <cntrl><D> will no longer be used to open drive D into the currently active window, instead it will be used to delete selected files.
Remember:
  Alternate + (A-P) = Open the drive's directory into a window
  Control   + (A-P) = Open the drive's directory in the active window
You can get all of the normal Desktop key commands by pressing <help> while at the Desktop.

Not only can you assign a key combo to "Delete," you can also remove the trash icon completely, and free up space for more important icons! Just select the Trash icon and then the menu item "Remove Icon." In fact, since TOS versions 2.0x and up allow you to open drives by pressing <alternate> and the drive letter, you can remove all the drive icons, too, and replace them with programs, folders, and files! If you need to actually see what's on drive D, just press <Alt><D>.

Something users of *any* TOS version can do is install more than one document type for an application! To do this, first install the application for one of the filetypes you want. Then load the DESKTOP/NEWDESK.INF file into an ASCII editor. Make a copy of the installed application line, making sure to keep all the copies together with the original in the .INF file. It will look something like:
   #Y FF 04 000 C:\path\GEMVIEW.GTP@ *.PI3@ @
Then rename the installed file type ("*.PI3") to whatever other file type you want. Make sure to keep all of these lines together. After doing this a few times, you might have the following lines in your .INF file:
   #Y FF 04 000 C:\path\GEMVIEW.GTP@ *.PI?@ @
   #Y FF 04 000 C:\path\GEMVIEW.GTP@ *.PC?@ @
   #Y FF 04 000 C:\path\GEMVIEW.GTP@ *.GIF@ @

Editing your INF file manually like this is the only way to install one application for more than one file type, since using "Install Application" from the desktop will just overwrite any previous assignment, rather than adding to it.

Of course, instead of installing an application for a particular extender, you might try editing the lines for a particular *prefix*! You could do something like:
   #G 03 04 000 C:\UTILS\FILE_VIE.WER\AV380.PRG@ READ*.*@ @
So that whenever you double clicked on a file like "READ.ME", ASCII View 3.80 would be run and the file loaded into it for viewing!

In fact, you can replace the Desktop's boring old [SHOW | PRINT | CANCEL] by installing an application for all file types. Be careful; since this works as a catch-all, you have to make sure that such an installation is the *last* line TOS checks when you double click. That means it has to be the first application line in the .INF file (remember, TOS uses the DESKTOP/NEWDESK.INF file as a sort of filter, working from the end of the .INF to the beginning. Basically, the

place to install an app like this is the line just before the *.APP, *.PRG, *.TOS, etc. is defined. See the annotated INF file below for an example.

NEAT HACKS

In any version of TOS you can, of course, change the name of any desktop icon to anything you want. This is true of the Trash can, which can be renamed HAZARD or whatever you want. It's also true of any programs or folders you put on the desktop (in TOS >= 2.0x); Just look at the line of the INF file where that file/folder appears. It will look something like:
#X 07 01 7B FF C:\UTILS\VIEWERS\VIEWER.PRG@ VIEWER.PRG@
The name seen on the desktop will be "VIEWER.PRG." You can change this to "SEE IT!" just by changing the line to something like:
#X 07 01 7B FF C:\UTILS\VIEWERS\VIEWER.PRG@ SEE IT!@

Of course, as I said above, you don't really need the trash or drive icons on TOS >= 2.0x, so why not just get rid of them? You can free up tons of space on the desktop for commonly used programs, files, and folders. It also has the nice side-affect of freeing up space in your INF file, allowing you to have a few more lines of icon or application assignments. This is a great tip for people who use Desk Switch or Load Inf; you can replace the drive icons with INF file icons, allowing you easy access not to your drives but to the work you need to do.

One feature that deserves to be recognized, though it isn't a hack, is the drag and drop feature of TOS >= 2.0x. With this feature, you can put your commonly used programs on the desktops and then just drag the file you want to work on over the app (until the app is highlighted), then "drop" the file. This has the advantage of requiring fewer steps to get working, as well as not requiring you to install the application in the INF file, thereby saving a few bytes of space.

Create INF files specific to programs, put them all in a common folder, and use Desk Switch or Load Inf to switch between them. You can have a DTP.INF, WORDPROC.INF, GEMDRAW.INF, DEGAS.INF, TELECOM.INF, UNARC.INF, etc. This frees up a lot of space in any particular INF file. And with each INF, each type of file used with that task can have its own unique icon, so it's easy to distinguish file types, and you can also have all the installed application info you want! Using just one INF file really limits you in what you can do, so make more!

Reset your keyboard-equivalents for the desktop menus so they make sense! Use normal keys for one menu, and <cntrl>-key combos for the other. You can even try <shft><cntrl>-key combos. Require two keys to be pressed when deleting or formating.

Get a resource editor (like MKRSC; this works with TOS 4.0x icons) and build your own icons, then assign them by hex number to files in your

INF file.

Put a folder on the the desktop and save the INF file. Then edit the INF file so that the line containing that folder begins with #X instead of #V. Now when you double click on the folder, the default document displayer is run, and the contents of the folder are loaded in a batch mode. This is an excellent way of viewing new pictures or text files! You can even use masks to control what sort of files load in. Of course, your viewer has to be capable of processing batch jobs.

Make yourself a blank icon (no mask, nothin'). You can now assign short notes to the name of the icon, such as Fkey assignments, short reminders, and whatnot. If you have TOS >= 2.0x, you can put these "sticky notes" on the desktop, too. They are easily editable from the desktop.

INF FILE COMPOSITION

Below is a (fairly) complete explanation of the lines in the DESKTOP.INF or NEWDESK.INF files. Some parameters only apply to specific versions of TOS. Compare what's below to what is already in your .INF file, and if it's not already there, DON'T ADD IT! TOS expects certain lines to have a particular format, and adding to these lines can confuse TOS and give you headaches. On the other hand, deleting things can have the same effect, so don't do that either.

Where a description is given by "bits," the following procedure will produce the needed hexidecimal number:

> start with the high bit (e.g. bit "7"), and begin writing down the desired configuration as a binary number (1's and 0's) from left to right. Unlisted bits should not be changed. When the number is complete, convert it to hex. Any decent calculator will do this without a fuss.
> For example: For the "Other configuration parameters" (see below), I want bit 4 set to "filname," so I write a 0; bit 3 I want set to "top window", so to the right of the zero I write another 0; then bit 2 I set "size to fit" on, so I write a 1 to the right of bit 3; bit 1 is not listed so I write a 1 (since that's what it was originally); finally bit 0 I want set to "sort on," so I write a 0. The resulting number is "00110." I set my calculator to "bin" and plug this number in, then convert it to hex (by changing the calculator mode to "hex"). The resulting number, which goes in the fourth column of line #E is 06.

If you aren't familiar with doing this, I suggest you try it out a couple times and compare it with what's already in your desktop file. Don't try changing anything until you are confident you can get it right! Wrong values can do unpredictable things to your system,

including causing it to crash!

I have been unable to determine the usage of some bits. In these cases, I list the bit with a question mark and give the current value in my NEWDESK.INF file. Yours may be different, so check it and use whatever your INF file uses. To check it, convert the current hex value to binary: the rightmost number is bit 0, the digit to the left of that is bit 1, and so on. Converting hex 06 to bin is 110: bit 0 = 0; bit 1 = 1, bit 2 = 1, bit 3 = 0 (not shown), and bit 4 = 0 (also not shown).

And remember, if you find you've made a mistake, just delete the broken .INF file and re-load the old version that you backed up.

All "@" symbols must be followed by a space.

==========

Keyboard equivalents (available on TOS >= 2.0x)

#K xx xx xx xx ... @
  next 28 columns = hexidecimal value of kbd equiv. (00 = none)
  next 2 columns = 00
  next column = Video prefs. key (not on all TOS versions)
  last column = @

==========

Video settings (some features not available on some versions of TOS)

#E PR BR xx OP LD CM xx xx xx...

   PR = Preferences (Confirm file copy, delete, and overwrite; display options)

     bit 7: show files as
        1 = show as text   0 = show as icons

     bits 6,5: sort
        00 = by name   01 = by date
        10 = by size   11 = by type

     bit 4: confirm deletes
        1 = yes   0 = no

     bit 3: confirm copy
        1 = yes   0 = no

     bit 2: ?
        0

     bit 1: ?
        0

```
  bit 0: confirm overwrites
     0 = yes                  1 = no

B = Blitter
     1 = on                   0 = off

R = Resolution #

     1 = ST Low  or RGB 320 x 200
     2 = ST Med  or RGB 640 x 200
     3 = ST Hi   or RGB 640 x 400
     4 = Falcon 80 col x 240 or 480
     5 = TT monochrome(?)
     6 = Falcon 40 col x 240 or 480

OP = Other configuration parmaters

  bit 4: Input parameter (from "Desktop Config")
     0 = filename             1 = pathname

  bit 3: Default directory (from "Desktop Config")
     0 = Top Window           1 = Application

  bit 2: Size to fit
     0 = off                  1 = on

  bit 1: ?
     1

  bit 0: Sort on, off
     0 = Sort on              1 = Sort off

 LD = Line Doubling or Interlace

     00 = Line Doubling or Intlace off
     01 = Line Doubling or Intlace on

 CM

  bit 7 - ST Compatibility:
     0 = non-compatible;      1 = ST compatibility on

  bit 4: Hardware Select
     0 = RGB mode             1 = VGA mode

  bit 3: nuber of text columns
     0 = 40 columns           1 = 80 columns

  bits 0-2: number of available colors
     000 = 2 colors
     001 = 4
     010 = 16
```

```
        011 = 256
        100 = Truecolor mode

================================================

Desktop & Window settings

#Q xx xx xx xx DC WB

    D = Desktop Background Pattern

        0 = transparent             1 = Pattern #1
        2 = Pattern #2     ...      7 = Solid

    C = Desktop background color

        0 = Color #1                1 = Color #2
        A = Color #10      ...      F = Color #16

    W = Window Background Pattern

        0 = transparent             1 = Pattern #1
        2 = Pattern #2     ...      7 = Solid

    B = Window background color

        0 = Color #1                1 = Color #2
        A = Color #10      ...      F = Color #16

=====================================================================

Installed Applications have the following info

   #n 04 04 000 C:\path\progname@ *.*@ @

Where #n is any of the following:
     #G  for GEM prg
     #Y  for GTP prg
     #P  for TTP prg
     #F  for TOS prg

The first 04 is the prg icon.  Changing this has no effect
The second 04 is the document icon.  Changing this has no effect.
Both of these should be set to the same value.

  000  contains the following information :

       First Digit  -   Number           Meaning
                          0      Default Directory = Top Window
                                 Parameter = Filename Only
                          1      Default Directory = Application
                                 Parameter = Filename Only
                          2      Default Directory = Top Window
```

```
                                Parameter = Full Path
                           3    Default Directory = Application
                                Parameter = Full Path

       Second and Third Digit  =  Function key assignment F1 to
       shiftF10:
                  00 = no assignment      01 = F1
                  0A = F10                0F = F15
                  10 = F16                14 = F20
```

Following this is the full path and file name, followed immediately by an "@" symbol. After the "@" is a space followed by the installed document type (if any), followed by a second "@". Next comes another space followed by any parameters that are to be passed to the program (for GTP or TTP programs). Finally comes another "@" and a space.

For example,

```
  #P FF 04 009 C:\PATH\VIEWER.TTP@ *.*@ @
```

would install the program VIEWER.TTP as the default document displayer (this then supercedes the desktop [SHOW | PRINT | CANCEL]). This line also sets VIEWER.TTP to run when F9 is pressed. There are no parameters passed to this program.

==========

Default window icons-

```
 Example: #G 03 03 000 @ *.PRG@ @
```

First column = type
- #I = specific file or file type *
- #N = non-specific file *
- #D = folder
- #G = Gem app
- #Y = GTP app
- #P = TTP app
- #F = Tos app

second column = default icon # for #G, #F, #Y, #P
- New default icon can be installed for app type by changing this #

third column = default icon # for #N, #D
- New icon can be installed for non-prg files by changing this #

Both the second and third column should contain the same hexidecimal number.

fourth column - function key assignment. Leave at 000 (no assingment)

fifth column - @ filename.ext@ @ wildcard

```
* #I assigns an icon to the specified file or file type.  It does not
  assign an action.
  #N assigns an icon to the specified file or file type and specifies
  it as a non-executable file.  If the line
      #N 0B 0B 000 @ *.*@ @
  appears below the default lines for APP, PRG, etc. (i.e.
      #G 03 FF 000 *.APP@ @ @
      #G 03 FF 000 *.PRG@ @ @ etc.),
  programs will not execute when double-clicked on.

=====================================================================

Desktop Icons-

 Example: #X 08 02 18 FF   C:\UTILS\VIEWER\VIEWER.PRG@ SEE IT!@

first column = type
    #M = Storage Device
    #T = Trash
    #O = Printer
    #X = Application
    #V = Directory

second column = horizontal position in icon widths

third column = vertical position in icon widths

fourth column = Hex value of Icon number in DESKTOP.RSC

fifth column = FF (function unknown)

sixth column = Device ID (A-P) or space

Finally: full path and filename@ icon label@  (for a file or folder)
         Device name@ @                       (for a device)

=====================================================================



Annotated NEWDESK.INF File:
---------------------------

#a000000
Stores serial communication info.  Not very important since
the terminal program alters this info anyway.

#b000000
Printer Configuration:
   digit \ setting      0              1
     1              dot matrix      daisy wheel
     2            black & white       color
     3              1280 dpl         960 dpl
```

```
     4                draft           final
     5              parallel         serial
     6             continuous      single sheet

#c777000700060007005520050555222077055707505550770311110 3
Color palette settings, mouse double-click response, key-click, bell
sound, key delay and key repeat rate.

#d
Reserved.  Needs to have roughly forty-seven spaces since GEM looks
roughly 124 bytes into the file for the next line.

#Z 01 F:\RUNPROG\PROGRAM.PRG@
Program to Auto-run at bootup.  Does not apply to TOS earlier than
1.04.

#K 4F 49 53 1F 46 20 43 00 41 4D 00 09 14 0E 04 13 05 0F 00 00 00 01
00 12 00 52 00 44 00 00 @
Keyboard assignments for desktop menu items. (should all be on one
line)

#E 50 13 00 06
Default screen resolution; window sorting method; show as icon
or text; confirm copies, deletes, etc.  Second digit of the second
byte is for screen resolution: 13 = blitter on, ST High rez.

#Q 41 00 43 40 43 40

#W 00 00 02 07 4B 11 00 F:\RUNPROG\*.PRG@
#W 00 00 16 02 35 0B 00 @
#W 00 00 2E 0E 22 0A 00 @
#W 00 00 00 0C 21 0C 00 @
#W 00 00 00 04 4C 0C 00 @
#W 00 00 02 0D 4C 09 00 @
#W 00 00 0E 0F 2A 09 00 @
#W 00 00 06 01 34 09 00 @
Default window locations, sizes, and open directories and masks.

#P 03 04 000 C:\UTILS\FILE_VIE.WER\GUCK18\GUCK.TTP@ *.*@ @
This line installs Guck as the default file viewer.  When the file is
double clicked on, TOS checks all the #G, #P, #Y, and #F assignments
below, and if the file doesn't match any of those, Guck is run and the
file is loaded in.  Notice that it is the first assignment in the INF
file, and hence the last assigment TOS checks.  If you moved this one
down a few lines (say, underneath #F 03 04 000 *.TOS@ @ @ ), you'd
never get any programs to run; everything (except the .INFs and
archives) would be loaded directly into Guck!

#G 03 FF 000 *.ACC@ @ @
 - This line lets you run an ACC as a program, if the ACC is capable of it,
 by double-clicking on the ACC in any window.

#G 03 FF 000 *.APP@ @ @
```

```
#G 03 FF 000 *.PRG@ @ @
#Y 03 FF 000 *.GTP@ @ @
#P 03 FF 000 *.TTP@ @ @
#F 03 04 000 *.TOS@ @ @
#G 03 A0 200 C:\UTILS\ARC\STZIP26\STZIP.PRG@ *.ZIP@ @
#G 03 2F 200 C:\UTILS\ARC\TWOINONE.PRG@ *.ARC@ @
#G 03 2F 200 C:\UTILS\ARC\TWOINONE.PRG@ *.LZH@ @
#G 03 2F 200 C:\UTILS\ARC\TWOINONE.PRG@ *.ZOO@ @
#G 03 2F 200 C:\UTILS\ARC\TWOINONE.PRG@ *.ARJ@ @
#G 03 2F 200 C:\UTILS\LOADINF\LOADINF.PRG@ *.INF@ @
Executable programs and programs to run when files with the
appropriate mask are opened.  The second two digits of the third
column of numbers ("00" of the "200") is the hex number of the
function key assignment.  "0A" would be F10.
Other parameters (for GTP or TTP) can be included before the last "@".

#D 0A 0A 000 @ *.*@ @
Default folder icon

#I 0B 0B 000 @ *.*@ @
Default file icon

#I 0C 0C 000 @ *.PR?@ @
#I 11 11 000 @ *.AC?@ @
#I 2E 2E 000 @ *.CPX@ @
#I 3A 3A 000 @ *.TTP@ @
#I 0D 0D 000 @ *.TOS@ @
#I 39 39 000 @ *.APP@ @
#I 39 39 000 @ *.PRG@ @
#I 12 12 000 @ *.ACC@ @
#I 52 52 000 @ READ*.*@ @
#I 52 52 000 @ *.ME@ @
#I 2C 2C 000 @ *.INF@ @
#I 13 13 000 @ *.RSC@ @
#I 3B 3B 000 @ DC*.*@ @
#I 1B 1B 000 @ *.HLP@ @
#I 6F 6F 000 @ G+*.*@ @
#I 3C 3C 000 @ DATADIET.*@ @
#I 46 46 011 @ WORDUP.PRG@ @
#I 49 49 004 @ QVIEW.PRG@ @
#I 51 51 000 @ SUPER*.PRG@ @
#I 63 63 002 @ PGS2.PRG@ @
#I 70 70 000 @ WARP9_ST.PRG@ @
#I 73 73 000 @ DFORMAT.PRG@ @
#I 3D 3D 00A @ SQUISHII.APP@ @
#I 6D 6D 001 @ TWOINONE.PRG@ @
File icons and function key assignments.  Notice these act as filters
also, and hence files are applied from the bottom up.  For example,
SQUISHII.APP is assigned icon 3D, but a file called SQUID.GHU doesn't
match any filter, it would be assigned the default icon 0B.

#D 4E 4E 000 @ TELECOMM@ @
#D 3E 3E 000 @ AUTO@ @
```

```
#D 37 37 000 @ CLIPBRD@ @
#D 53 53 000 @ ACCS@ @
#D 55 55 000 @ CPX@ @
#D 35 35 000 @ TEXTE@ @
#D 36 36 000 @ BILDER.PAD@ @
#D 36 36 000 @ MYDRAW@ @
#D 36 36 000 @ META@ @
#D 36 36 000 @ GRAPHICS.Z@ @
#D 4F 4F 000 @ GAMES@ @
Folder icon assignments.  Again, assignment is from bottom to top.

#M 00 00 00 FF A FLOPPY DISK@ @
#M 01 00 02 FF C BOOT@ @
#M 02 00 02 FF D PUBLISHING@ @
#M 03 00 02 FF E FILES@ @
#M 04 00 02 FF F PROGRAMMING@ @
#M 05 00 02 FF G GRAPHICS@ @
#M 06 00 02 FF H TELECOM@ @
#M 05 01 05 FF M RAM DISK@ @
Drive icon, letter, position, and label assignments.

#O 06 01 08 FF   PRINTER@ @
Printer icon, position, and label assignment.

#T 07 00 09 FF   GARBAGE@ @
Trash icon, position, and label assignment.

#X 07 01 7B FF   C:\UTILS\RECV_CAN\RT.PRG@ TRASH CAN@
Executable file on the desktop.  Icon, position, and label assignment.

#V 04 01 3E FF   C:\AUTO\SUPERBT\*.INF@ DESKTOP@
Directory folder on the desktop.  Position, icon, pathname, and label
assignment.


COMMENTS
--------
   Notice if a file is clicked on and is not a *.INF or one of the
listed archival types, TOS then checks to see if it is either TOS,
TTP, GTP, PRG, or APP (in that order), and if so executes it.  If it
is not one of these, the remaining installation for types *.*
"catches" it, so GUCK  will be run and the file will be loaded in.
This then replaces the  desktop [ SHOW | PRINT |CANCEL].  Remember
that each of these acts as a filter and that TOS applies them from
*bottom* to top; so when you double-click on a file, TOS first checks
to see if it is *.INF, and if so runs LOADINF.PRG, then it checks for
type *.ARJ, then... through the executables (TOS, TTP, GTP, PRG, APP
in that order), and finally if the file wasn't one of those, TOS type
*.* (that's everything left) is loaded into GUCK.


******************************END OF FILE**************************
```

--
Annius V. Groenink | avg@cwi.nl | Private/Edith/ZFC:
CWI, Kruislaan 413 | A.V.Groenink@zfc.nl | P.O. Box 12079
1098 SJ Amsterdam | Room M233 ext. 4077 | NL 1100 AB Amsterdam
The Netherlands | Phone: +31 20 592 4077 | Phone: +31 20 695 9901

From 2995914511650913297
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,84120fa177ff61e0,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,84120fa177ff61e0,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1994-11-29 08:46:29 PST
Newsgroups: comp.sources.atari.st,comp.sys.atari.st
Path: bga.com!news.sprintlink.net!EU.net!sun4nl!cwi.nl!avg
From: avg@cwi.nl (Annius Groenink)
Subject: v08INF4: Introduction to comp.sources.atari.st
Message-ID: <D01F6D.1JJ@cwi.nl>
Originator: avg@mijt.cwi.nl
Sender: news@cwi.nl (The Daily Dross)
Nntp-Posting-Host: mijt.cwi.nl
Organization: CWI, Amsterdam
Date: Tue, 29 Nov 1994 16:30:13 GMT
Approved: avg@cwi.nl (Annius Groenink)
Expires: Mon, 19 Dec 1994 16:30:13 GMT
Lines: 170
Xref: bga.com comp.sources.atari.st:18 comp.sys.atari.st:21806

Submitted-by: avg@cwi.nl (Annius Groenink)
Posting-number: Volume 8, Info 4

This article is posted twice a month. It gives a short
introduction to comp.sources.atari.st and describes how to submit
sources or digressions to the group.

I am always looking for suggestions on how to improve the usefulness of
the newsgroup, and can be contacted at the following internet address:

   avg@cwi.nl

Annius Groenink, Moderator.

                    ABOUT comp.sources.atari.st

The moderated usenet group comp.sources.atari.st contains publications
or announcements of sources and texts about programming for Atari TOS
computers. The publications on comp.sources.atari.st may be freely

distributed as long as modifications are documented and the original authors are mentioned.

THIS IS A MODERATED GROUP. YOU CANNOT POST TO IT BY YOURSELF. Read the instructions below for submitting material intended for publication on comp.sources.atari.st.

SUBMITTING SOURCES OR DIGRESSIONS FOR PUBLICATION

Items intended for posting or queries and problem notes should be sent to avg@cwi.nl. If you are on a UUCP-only site, you can send them to uunet!cwi.nl!avg.

If you want verification of arrival, say so at the beginning of your submission, or in a cover note. I try to verify that a program works, and if I can't get it to work, I may hold up posting it for a couple of days. Please note that, except in rare cases, software without documentation will not be published. The backlog from receipt to posting varies from one to four weeks depending mostly on the set of submissions currently in my queue.

If you are submitting both sources and binaries, PLEASE send the two separately. If I have to separate your sources from your binaries by hand, your submission will most likely sit on the back burner for a while, or be rejected outright.

Sources can be submitted in ASCII form, but for large pacages, a ZOO 2.1 archive encoded using Dumas uuencode is preferred.

Please include a description of your program at the top of your article. Look at old issues of comp.binaries.atari.st for examples of the sorts of descriptions I'm looking for. In general you should indicate what the program does, on what types of Atari TOS computers, whether your program requires a certain amount of on-board memory, a hard disk or a certain screen resolution. If necessary, I may rewrite or add to this information in order to make the postings to the group more uniform. I can't check your submission in all possible configurations, and see what works and what doesn't. Without a description, your submission is likely to be rejected.

THE STRUCTURE OF COMP.SOURCES.ATARI.ST ARTICLES

There are two types of articles in comp.binaries.atari.st: "information postings" and "sources". The source postings are also called "Issues". For a while, 'announcements' were also accepted, but there is now a dedicated group, comp.sys.atari.announce, for that purpose. The type of a posting can be distinguished by the subject line:

```
  Subject: v02INF1: Introduction to comp.sources.atari.st
```

This first word in the title identifies this as the first info posting of volume two. Similarly, the subject line shown below:

```
  Subject: v06i081: godsrc -- Universe simulator
```

identifies this as the 81st source article in Volume 6. Large sources are broken up into smaller pieces, and have subject lines that look like this:

```
  Subject: v06i088: decbustr -- VMS clone in ST-LOGO, Part03/08
```

Certain information about the system configuration required to use the program is given on the keywords line.

```
  Keywords: MWC, Megamax, 1meg, high
```

This means that the program compiles with Mark Williams or Megamax C, requires at least one meg of RAM and only runs in high resolution. Following is a list of keywords; new ones may be added as needed. They are mostly self- explanatory.

```
  Alcyon  - Alcyon C (included with Atari developer's kit)
  MWC         - Mark Williams C
  Megamax - Megamax C
  Laser       - Laser C
  Lattice - Lattice C
  C       - Any C compiler
  OSS         - OSS Personal Pascal
  BASIC       - ST BASIC (this is not a joke!)
  uuencode    - program is uuencoded (uudecode required to unpack)
  arc         - program is archived (arc required to unpack)
  zoo-high    - program is archived (Zoo 2.1 required to unpack)
  lharc       - program is archived (lharc required to unpack)
  high        - high resolution
  medium  - medium resolution
  low         - low resolution
  1meg        - needs 1 meg of RAM
```

The References: line, if present, points to the previous part of a multi-part program, so that newsreaders (trn, nn, etc.) which can follow threads will recognize a large program as a single thread.

The first few lines of an article are auxiliary headers that look like this:

```
  Submitted-by: david@bdt.UUCP (David Beckmeyer)
  Posting-number: Volume 6, Issue 80
  Archive-name: micro-rtx
```

The "Submitted by" is the author of the program. If you have comments about the sources published in comp.sources.atari.st this is the person to contact. When possible, this address is in domain form, otherwise it is a UUCP bang path relative to some major (backbone) site.

The second line repeats the volume/issue information for the aide of notes sites and automatic archiving programs.

The Archive-name is the "official" name of this source in the archive. Large postings will have names that look like this:

Archive-name: desktop/part01

REPORTING AND TRACKING BUGS AND PATCHES TO POSTINGS

Most of the time, bugs and updates will be posted to comp.sys.atari.st. Sometimes, "official" updates from the author will be posted submitted to comp.sources.atari.st so that they are archived with the sources.

Since there is no easy way to apply source code patches on the ST, changes to a program often result in a whole new version being posted.

To report bugs, contact the person listed in the Submitted-to header. Often there is a contact address in a README file, too. I do not maintain the sources I moderate, so don't send your bug reports to me.

HOW TO OBTAIN OLD ISSUES

There is currently no officially documented archive for this group.

--
Annius V. Groenink | avg@cwi.nl | Private/Edith/ZFC:
CWI, Kruislaan 413 | A.V.Groenink@zfc.nl | P.O. Box 12079
1098 SJ Amsterdam | Room M233 ext. 4077 | NL 1100 AB Amsterdam
The Netherlands | Phone: +31 20 592 4077 | Phone: +31 20 695 9901

From -3269455058774213238
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,cdce305677a285b,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1995-02-09 02:00:58 PST
Path: swrinde!hookup!newshost.marcam.com!zip.eecs.umich.edu!panix!not-for-mail
From: ccapc@cyber.sell.com (Consumer Credit Advocates)
Newsgroups: comp.sources.atari.st
Subject: <ad> GUARANTEED CREDIT REPAIR BY LAW FIRM
Date: 9 Feb 1995 05:00:50 -0500

Organization: Consumer Credit Advocates, PC
Lines: 117
Sender: ccapc@panix.com
Approved: postmaster
Message-ID: <3hcp4i$gm4@panix.com>
NNTP-Posting-Host: panix.com

Consumer Credit Advocates, PC
11 Pennsylvania Plaza, Suite 2101
New York, NY 10001
(212) 629-5261 (telephone) (212) 629-4762 (fax)
E-MAIL: ccapc@cyber.sell.com

Our LAW FIRM offers direct guaranteed effective credit restoration services by experienced attorneys. THIS IS NOT A DO-IT-YOURSELF KIT.

What can we do?

We have successfully facilitated the removal of Late Payments, Charge-offs, Foreclosures, Repossessions, Collection Accounts, Loan Defaults, Tax Liens, Judgments and Bankruptcies from our clients' credit reports. WE GUARANTEE THAT YOUR CREDIT CAN BE RESTORED!!!

Who needs our services?

Anyone who has experienced the inconvenience and embarrassment of being turned down for a credit card, a lease or a purchase of an automobile.

Anyone who is unable to buy the house of their dreams due to denial of a mortgage application or who has to pay thousands of dollars more in mortgage interest than someone with good credit.

Anyone who has been turned down for a job or promotion due to derogatory credit items on a credit report.

Anyone in business who has lost a deal because a person or firm wanted to see his/her credit report before doing business.

Anyone who has been unable to establish credit.

THE FOUR GREAT MYTHS OF CREDIT:

Myth #1: It is illegal or immoral to have your credit report cleared.

> Fact: It is not illegal nor immoral. In fact, that is what the Fair Credit Reporting Act is all

about. The act was enacted by Congress in 1971. One of its purposes as to give consumers the protection of the law and to help guard against any unwarranted invasion of a consumer's right to privacy.

Myth #2: The information on a credit report cannot be changed.

Fact: Actually, the opposite is true under the Fair Credit Reporting Act. Federal and State laws require that items be removed if they are not 100% accurate or cannot be verified in a timely manner.

Myth #3: It is impossible to get a bankruptcy off a credit report.

Fact: Bankruptcies can come off credit reports like any other derogatory item. The nature of the derogatory item has nothing to do with its removal under the Fair Credit Reporting Act.

Myth #4: Credit Reporting Agencies are empowered with some kind of governmental authority.

Fact: Absolutely Not!! They are simply large corporations whose primary goal is to make a profit like any other business.

If you have ever applied for or received credit, a file exists in one or more of the credit bureaus. These companies collect, store and distribute as much credit information as they can find, retaining negative information on a credit report for 7 to 10 years. This information is evaluated by potential creditors to determine your credit worthiness.

Credit reporting agencies are in business to protect the interests of the creditors. the LAW FIRM's goal is to help and protect the individual consumer from inaccurate credit reporting.

The president of our LAW FIRM has been practicing consumer law since 1984. The staff of our firm has successfully processed, disputed and challenged thousands of client credit reports.

Our legal fee is based o the number of negative items that appear on a client's credit reports, issued b the three national credit bureaus. Our retainer agreement offers a MONEY BACK GUARANTEE stating that if any

negative item is not deleted, upgraded or corrected from a client's credit file, it will give the client a full refund for that item or continue to process the client's file at no additional fee until that item is corrected, upgraded or deleted. THE ONLY THING YOU HAVE TO LOSE IS YOUR BAD CREDIT!!

PLEASE CONTACT THE LAW FIRM AND LEAVE YOUR FULL NAME, MAILING ADDRESS AND TELEPHONE NUMBER SO WE MAY FORWARD FURTHER INFORMATION AND INSTRUCTIONS TO YOU ABOUT OUR SERVICE.

Consumer Credit Advocates, PC
11 Pennsylvania Plaza, Suite 2101
New York, NY 10001
(212) 629-5261 (telephone) (212) 629-4762 (fax)
E-MAIL: ccapc@cyber.sell.com

From -5397302508240358130
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 1157bf,9a4731d766dc15fc,start
X-Google-Attributes: gid1157bf,public
X-Google-Thread: 10d527,9a4731d766dc15fc,start
X-Google-Attributes: gid10d527,public
X-Google-Thread: 100493,9a4731d766dc15fc,start
X-Google-Attributes: gid100493,public
From: 2081833.42663942(Your Lifetime Health Planner)
Subject: The Key To Organizing Your Health Records
Date: 1995/05/24
Message-ID: <3pujj9$k2a@baygull.rtd.com>#1/1
X-Deja-AN: 103127131
approved: trasoff@rtd.com
sender: 2081833.42663942 (Your Lifetime Health Planner)
news-posting-software: Cyberspam
organization: Your Lifetime Health Planner
newsgroups: control,comp.sources.amiga,comp.sources.apple2,comp.sources.atari.st

THE KEY TO ORGANIZING YOUR HEALTH RECORDS

"If you care about yourself and your children, you will do everything you can to know your family's medical history. It can save your life, and the lives of those who are dear to you." -- Dr. Aubry Milunsky, Director, Boston University's Center for Human Genetics (Life Magazine, March, 1995).

Your Lifetime Health Planner offers the simplest, most effective way to take charge of your health. Enter important facts about your health history, keep a record of the names of medications and their effects, document each doctor visit and diagnosis. You will have the information you need, and in the event of an emergency, so will your family.

Your Lifetime Health Planner, 198 pages in a compact, loose-leaf binder, contains a variety of sections which include "Doctor Visits," "Medications," "Family Health History," and "Doctors and Specialists." The Planner is a must for the busy executive who needs to stay organized, for seniors who are receiving care and for newborns who need a record of their health care.

Parents Magazine recommended the Planner to its readers (April, 1994) and MCI called it "a useful resource in managing your personal health records." Margie Smotherman, Coordinator, Employee Health and Wellness, Blue Shield/Blue Cross of Iowa, said: "The employees felt that these (Planners) were great tools to not only record their pertinent numbers, but to plan for doctor's visits."

To get your copy, send a check or money order for $19.95 (plus shipping and handling) to :

Your Lifetime Health Planner
1955 W Grant Rd., # 230
Tucson AZ 85745
USA

Shipping and handling charges: $4.95 US; $8 Canada/Mexico; $14 Europe; $18 Asia/Pacific Rim.

Include your name, address and telephone number.*

Visa/MasterCard orders, call 520-798-1530.

*Large corporate discounts available for orders of 1,000 copies or more.

Your Lifetime Health Planner
1955 West Grant Road, #230
Tucson, AZ 85747

Enclosed is $_____ for _____ copies of Your Lifetime Health Planner.

Please ship to:

YOUR NAME _____

YOUR ADDRESS _____
Street
_____
City State Zip or Postal Code Country

YOUR TELEPHONE NUMBER _____

From -7833508864784538929
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,6eb416b52bd7bc53,start

X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,6eb416b52bd7bc53,start
X-Google-Attributes: gid100493,public
From: avg@cwi.nl (Annius Groenink)
Subject: v08INF5: Introduction to comp.sources.atari.st
Date: 1995/07/13
Message-ID: <DBnK8x.DFr@cwi.nl>
X-Deja-AN: 106177991
expires: Wed, 2 Aug 1995 11:36:32 GMT
approved: avg@cwi.nl (Annius Groenink)
sender: news@cwi.nl (The Daily Dross)
organization: CWI, Amsterdam
newsgroups: comp.sources.atari.st,comp.sys.atari.st
originator: avg@mijt.cwi.nl

Submitted-by: avg@cwi.nl (Annius Groenink)
Posting-number: Volume 8, Info 5

This article is posted twice a month. It gives a short introduction to comp.sources.atari.st and describes how to submit sources or digressions to the group.

I am always looking for suggestions on how to improve the usefulness of the newsgroup, and can be contacted at the following internet address:

   avg@cwi.nl

Annius Groenink, Moderator.

ABOUT comp.sources.atari.st

The moderated usenet group comp.sources.atari.st contains publications or announcements of sources and texts about programming for Atari TOS computers. The publications on comp.sources.atari.st may be freely distributed as long as modifications are documented and the original authors are mentioned.

THIS IS A MODERATED GROUP. YOU CANNOT POST TO IT BY YOURSELF. Read the instructions below for submitting material intended for publication on comp.sources.atari.st.

SUBMITTING SOURCES OR DIGRESSIONS FOR PUBLICATION

Items intended for posting or queries and problem notes should be sent to avg@cwi.nl. If you are on a UUCP-only site, you can send them to uunet!cwi.nl!avg.

If you want verification of arrival, say so at the beginning of your submission, or in a cover note. I try to verify that a program works, and if I can't get it to work, I may hold up posting it for a couple of days. Please note that, except in rare cases, software without documentation will not be published. The backlog from receipt to posting varies from one to four weeks depending mostly on the set of submissions currently in my queue.

If you are submitting both sources and binaries, PLEASE send the two separately. If I have to separate your sources from your binaries by hand, your submission will most likely sit on the back burner for a while, or be rejected outright.

Sources can be submitted in ASCII form, but for large pacages, a ZOO 2.1 archive encoded using Dumas uuencode is preferred.

Please include a description of your program at the top of your article. Look at old issues of comp.binaries.atari.st for examples of the sorts of descriptions I'm looking for. In general you should indicate what the program does, on what types of Atari TOS computers, whether your program requires a certain amount of on-board memory, a hard disk or a certain screen resolution. If necessary, I may rewrite or add to this information in order to make the postings to the group more uniform. I can't check your submission in all possible configurations, and see what works and what doesn't. Without a description, your submission is likely to be rejected.

THE STRUCTURE OF COMP.SOURCES.ATARI.ST ARTICLES

There are two types of articles in comp.binaries.atari.st: "information postings" and "sources". The source postings are also called "Issues". For a while, 'announcements' were also accepted, but there is now a dedicated group, comp.sys.atari.announce, for that purpose. The type of a posting can be distinguished by the subject line:

```
Subject: v02INF1: Introduction to comp.sources.atari.st
```

This first word in the title identifies this as the first info posting of volume two. Similarly, the subject line shown below:

```
Subject: v06i081: godsrc -- Universe simulator
```

identifies this as the 81st source article in Volume 6. Large sources are broken up into smaller pieces, and have subject lines that look like this:

```
Subject: v06i088: decbustr -- VMS clone in ST-LOGO, Part03/08
```

Certain information about the system configuration required to use the program is given on the keywords line.

```
Keywords: MWC, Megamax, 1meg, high
```

This means that the program compiles with Mark Williams or Megamax C, requires at least one meg of RAM and only runs in high resolution. Following is a list of keywords; new ones may be added as needed. They are mostly self- explanatory.

```
Alcyon  - Alcyon C (included with Atari developer's kit)
MWC         - Mark Williams C
Megamax - Megamax C
Laser       - Laser C
Lattice - Lattice C
C       - Any C compiler
OSS         - OSS Personal Pascal
BASIC       - ST BASIC (this is not a joke!)
uuencode    - program is uuencoded (uudecode required to unpack)
arc         - program is archived (arc required to unpack)
zoo-high    - program is archived (Zoo 2.1 required to unpack)
lharc       - program is archived (lharc required to unpack)
high        - high resolution
medium  - medium resolution
low         - low resolution
1meg        - needs 1 meg of RAM
```

The References: line, if present, points to the previous part of a multi-part program, so that newsreaders (trn, nn, etc.) which can follow threads will recognize a large program as a single thread.

The first few lines of an article are auxiliary headers that look like this:

```
Submitted-by: david@bdt.UUCP (David Beckmeyer)
Posting-number: Volume 6, Issue 80
Archive-name: micro-rtx
```

The "Submitted by" is the author of the program. If you have comments about the sources published in comp.sources.atari.st this is the person to contact. When possible, this address is in domain form, otherwise it is a UUCP bang path relative to some major (backbone) site.

The second line repeats the volume/issue information for the aide of notes sites and automatic archiving programs.

The Archive-name is the "official" name of this source in the archive. Large postings will have names that look like this:

```
Archive-name: desktop/part01
```

REPORTING AND TRACKING BUGS AND PATCHES TO POSTINGS

Most of the time, bugs and updates will be posted to comp.sys.atari.st. Sometimes, "official" updates from the author will be posted submitted to comp.sources.atari.st so that they are archived with the sources.

Since there is no easy way to apply source code patches on the ST, changes to a program often result in a whole new version being posted.

To report bugs, contact the person listed in the Submitted-to header. Often there is a contact address in a README file, too. I do not maintain the sources I moderate, so don't send your bug reports to me.

HOW TO OBTAIN OLD ISSUES

There is currently no officially documented archive for this group.

--
Annius V. Groenink | avg@cwi.nl | Private/Edith/ZFC:
CWI, Kruislaan 413 | zfc@zfc.nl | P.O. Box 12079
1098 SJ Amsterdam | Room M233 ext. 4077 | NL 1100 AB Amsterdam
The Netherlands | Phone: +31 20 592 4077 | Phone: +31 20 695 9901

From -4973256148522018087
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,72523147a2238541,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,72523147a2238541,start
X-Google-Attributes: gid100493,public
From: avg@cwi.nl (Annius Groenink)
Subject: v08INF6: Introduction to comp.sources.atari.st
Date: 1995/07/31
Message-ID: <DCL0LH.J4A@cwi.nl>
X-Deja-AN: 107283326
expires: Sun, 20 Aug 1995 13:10:29 GMT
approved: avg@cwi.nl (Annius Groenink)
sender: news@cwi.nl (The Daily Dross)
organization: CWI, Amsterdam
newsgroups: comp.sources.atari.st,comp.sys.atari.st
originator: avg@mijt.cwi.nl

Submitted-by: avg@cwi.nl (Annius Groenink)
Posting-number: Volume 8, Info 6

This article is posted twice a month. It gives a short introduction to comp.sources.atari.st and describes how to submit

sources or digressions to the group.

I am always looking for suggestions on how to improve the usefulness of the newsgroup, and can be contacted at the following internet address:

   avg@cwi.nl

Annius Groenink, Moderator.

ABOUT comp.sources.atari.st

The moderated usenet group comp.sources.atari.st contains publications or announcements of sources and texts about programming for Atari TOS computers. The publications on comp.sources.atari.st may be freely distributed as long as modifications are documented and the original authors are mentioned.

THIS IS A MODERATED GROUP. YOU CANNOT POST TO IT BY YOURSELF. Read the instructions below for submitting material intended for publication on comp.sources.atari.st.

SUBMITTING SOURCES OR DIGRESSIONS FOR PUBLICATION

Items intended for posting or queries and problem notes should be sent to avg@cwi.nl. If you are on a UUCP-only site, you can send them to uunet!cwi.nl!avg.

If you want verification of arrival, say so at the beginning of your submission, or in a cover note. I try to verify that a program works, and if I can't get it to work, I may hold up posting it for a couple of days. Please note that, except in rare cases, software without documentation will not be published. The backlog from receipt to posting varies from one to four weeks depending mostly on the set of submissions currently in my queue.

If you are submitting both sources and binaries, PLEASE send the two separately. If I have to separate your sources from your binaries by hand, your submission will most likely sit on the back burner for a while, or be rejected outright.

Sources can be submitted in ASCII form, but for large pacages, a ZOO 2.1 archive encoded using Dumas uuencode is preferred.

Please include a description of your program at the top of your article. Look at old issues of comp.binaries.atari.st for examples of the sorts of descriptions I'm looking for. In general you should indicate what the program does, on what types of Atari TOS computers, whether your program requires a certain amount of on-board memory, a hard disk or a certain

screen resolution. If necessary, I may rewrite or add to this information in order to make the postings to the group more uniform. I can't check your submission in all possible configurations, and see what works and what doesn't. Without a description, your submission is likely to be rejected.

THE STRUCTURE OF COMP.SOURCES.ATARI.ST ARTICLES

There are two types of articles in comp.binaries.atari.st: "information postings" and "sources". The source postings are also called "Issues". For a while, 'announcements' were also accepted, but there is now a dedicated group, comp.sys.atari.announce, for that purpose. The type of a posting can be distinguished by the subject line:

```
Subject: v02INF1: Introduction to comp.sources.atari.st
```

This first word in the title identifies this as the first info posting of volume two. Similarly, the subject line shown below:

```
Subject: v06i081: godsrc -- Universe simulator
```

identifies this as the 81st source article in Volume 6. Large sources are broken up into smaller pieces, and have subject lines that look like this:

```
Subject: v06i088: decbustr -- VMS clone in ST-LOGO, Part03/08
```

Certain information about the system configuration required to use the program is given on the keywords line.

```
Keywords: MWC, Megamax, 1meg, high
```

This means that the program compiles with Mark Williams or Megamax C, requires at least one meg of RAM and only runs in high resolution. Following is a list of keywords; new ones may be added as needed. They are mostly self- explanatory.

```
Alcyon  - Alcyon C (included with Atari developer's kit)
MWC         - Mark Williams C
Megamax - Megamax C
Laser       - Laser C
Lattice - Lattice C
C       - Any C compiler
OSS         - OSS Personal Pascal
BASIC       - ST BASIC (this is not a joke!)
uuencode    - program is uuencoded (uudecode required to unpack)
arc         - program is archived (arc required to unpack)
zoo-high    - program is archived (Zoo 2.1 required to unpack)
```

```
  lharc       - program is archived (lharc required to unpack)
  high        - high resolution
  medium  - medium resolution
  low         - low resolution
  1meg        - needs 1 meg of RAM
```

The References: line, if present, points to the previous part of a multi-part program, so that newsreaders (trn, nn, etc.) which can follow threads will recognize a large program as a single thread.

The first few lines of an article are auxiliary headers that look like this:

```
  Submitted-by: david@bdt.UUCP (David Beckmeyer)
  Posting-number: Volume 6, Issue 80
  Archive-name: micro-rtx
```

The "Submitted by" is the author of the program. If you have comments about the sources published in comp.sources.atari.st this is the person to contact. When possible, this address is in domain form, otherwise it is a UUCP bang path relative to some major (backbone) site.

The second line repeats the volume/issue information for the aide of notes sites and automatic archiving programs.

The Archive-name is the "official" name of this source in the archive. Large postings will have names that look like this:

```
Archive-name: desktop/part01
```

REPORTING AND TRACKING BUGS AND PATCHES TO POSTINGS

Most of the time, bugs and updates will be posted to comp.sys.atari.st. Sometimes, "official" updates from the author will be posted submitted to comp.sources.atari.st so that they are archived with the sources.

Since there is no easy way to apply source code patches on the ST, changes to a program often result in a whole new version being posted.

To report bugs, contact the person listed in the Submitted-to header. Often there is a contact address in a README file, too. I do not maintain the sources I moderate, so don't send your bug reports to me.

HOW TO OBTAIN OLD ISSUES

There is currently no officially documented archive for this group.

--
Annius V. Groenink | avg@cwi.nl | Private/Edith/ZFC:
CWI, Kruislaan 413 | zfc@zfc.nl | P.O. Box 12079
1098 SJ Amsterdam | Room M233 ext. 4077 | NL 1100 AB Amsterdam
The Netherlands | Phone: +31 20 592 4077 | Phone: +31 20 695 9901

From 197368991041941325
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,72523147a2238541
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,72523147a2238541
X-Google-Attributes: gid100493,public
From: avg@cwi.nl (Annius Groenink)
Subject: v08INF6: Introduction to comp.sources.atari.st
Date: 1995/07/31
Message-ID: <DCL0MJ.J5B@cwi.nl>
X-Deja-AN: 107283328
expires: Sun, 20 Aug 1995 13:11:07 GMT
approved: avg@cwi.nl (Annius Groenink)
sender: news@cwi.nl (The Daily Dross)
organization: CWI, Amsterdam
newsgroups: comp.sources.atari.st,comp.sys.atari.st
originator: avg@mijt.cwi.nl

Submitted-by: avg@cwi.nl (Annius Groenink)
Posting-number: Volume 8, Info 6

This article is posted twice a month. It gives a short introduction to comp.sources.atari.st and describes how to submit sources or digressions to the group.

I am always looking for suggestions on how to improve the usefulness of the newsgroup, and can be contacted at the following internet address:

   avg@cwi.nl

Annius Groenink, Moderator.

ABOUT comp.sources.atari.st

The moderated usenet group comp.sources.atari.st contains publications or announcements of sources and texts about programming for Atari TOS computers. The publications on comp.sources.atari.st may be freely distributed as long as modifications are documented and the original authors are mentioned.

THIS IS A MODERATED GROUP. YOU CANNOT POST TO IT BY YOURSELF. Read the

instructions below for submitting material intended for publication on comp.sources.atari.st.

SUBMITTING SOURCES OR DIGRESSIONS FOR PUBLICATION

Items intended for posting or queries and problem notes should be sent to avg@cwi.nl. If you are on a UUCP-only site, you can send them to uunet!cwi.nl!avg.

If you want verification of arrival, say so at the beginning of your submission, or in a cover note. I try to verify that a program works, and if I can't get it to work, I may hold up posting it for a couple of days. Please note that, except in rare cases, software without documentation will not be published. The backlog from receipt to posting varies from one to four weeks depending mostly on the set of submissions currently in my queue.

If you are submitting both sources and binaries, PLEASE send the two separately. If I have to separate your sources from your binaries by hand, your submission will most likely sit on the back burner for a while, or be rejected outright.

Sources can be submitted in ASCII form, but for large pacages, a ZOO 2.1 archive encoded using Dumas uuencode is preferred.

Please include a description of your program at the top of your article. Look at old issues of comp.binaries.atari.st for examples of the sorts of descriptions I'm looking for. In general you should indicate what the program does, on what types of Atari TOS computers, whether your program requires a certain amount of on-board memory, a hard disk or a certain screen resolution. If necessary, I may rewrite or add to this information in order to make the postings to the group more uniform. I can't check your submission in all possible configurations, and see what works and what doesn't. Without a description, your submission is likely to be rejected.

THE STRUCTURE OF COMP.SOURCES.ATARI.ST ARTICLES

There are two types of articles in comp.binaries.atari.st: "information postings" and "sources". The source postings are also called "Issues". For a while, 'announcements' were also accepted, but there is now a dedicated group, comp.sys.atari.announce, for that purpose. The type of a posting can be distinguished by the subject line:

```
Subject: v02INF1: Introduction to comp.sources.atari.st
```

This first word in the title identifies this as the first info posting of volume two. Similarly, the subject line shown below:

```
  Subject: v06i081: godsrc -- Universe simulator
```

identifies this as the 81st source article in Volume 6. Large sources are broken up into smaller pieces, and have subject lines that look like this:

```
  Subject: v06i088: decbustr -- VMS clone in ST-LOGO, Part03/08
```

Certain information about the system configuration required to use the program is given on the keywords line.

```
  Keywords: MWC, Megamax, 1meg, high
```

This means that the program compiles with Mark Williams or Megamax C, requires at least one meg of RAM and only runs in high resolution. Following is a list of keywords; new ones may be added as needed. They are mostly self- explanatory.

```
  Alcyon  - Alcyon C (included with Atari developer's kit)
  MWC         - Mark Williams C
  Megamax - Megamax C
  Laser       - Laser C
  Lattice - Lattice C
  C       - Any C compiler
  OSS         - OSS Personal Pascal
  BASIC       - ST BASIC (this is not a joke!)
  uuencode    - program is uuencoded (uudecode required to unpack)
  arc         - program is archived (arc required to unpack)
  zoo-high    - program is archived (Zoo 2.1 required to unpack)
  lharc       - program is archived (lharc required to unpack)
  high        - high resolution
  medium  - medium resolution
  low         - low resolution
  1meg        - needs 1 meg of RAM
```

The References: line, if present, points to the previous part of a multi-part program, so that newsreaders (trn, nn, etc.) which can follow threads will recognize a large program as a single thread.

The first few lines of an article are auxiliary headers that look like this:

```
  Submitted-by: david@bdt.UUCP (David Beckmeyer)
  Posting-number: Volume 6, Issue 80
  Archive-name: micro-rtx
```

The "Submitted by" is the author of the program. If you have comments about the sources published in comp.sources.atari.st this is the person to contact. When possible, this address is in domain form, otherwise it is a UUCP bang path relative to some major (backbone) site.

The second line repeats the volume/issue information for the aide of notes sites and automatic archiving programs.

The Archive-name is the "official" name of this source in the archive. Large postings will have names that look like this:

Archive-name: desktop/part01

REPORTING AND TRACKING BUGS AND PATCHES TO POSTINGS

Most of the time, bugs and updates will be posted to comp.sys.atari.st. Sometimes, "official" updates from the author will be posted submitted to comp.sources.atari.st so that they are archived with the sources.

Since there is no easy way to apply source code patches on the ST, changes to a program often result in a whole new version being posted.

To report bugs, contact the person listed in the Submitted-to header. Often there is a contact address in a README file, too. I do not maintain the sources I moderate, so don't send your bug reports to me.

HOW TO OBTAIN OLD ISSUES

There is currently no officially documented archive for this group.

--

Annius V. Groenink | avg@cwi.nl | Private/Edith/ZFC:
CWI, Kruislaan 413 | zfc@zfc.nl | P.O. Box 12079
1098 SJ Amsterdam | Room M233 ext. 4077 | NL 1100 AB Amsterdam
The Netherlands | Phone: +31 20 592 4077 | Phone: +31 20 695 9901

From 2476791382599616959
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,612cd89340fe52b5,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,612cd89340fe52b5,start
X-Google-Attributes: gid100493,public
From: avg@cwi.nl (Annius Groenink)
Subject: v08INF7: Introduction to comp.sources.atari.st
Date: 1995/09/01
Message-ID: <DE8A15.FpB@cwi.nl>
X-Deja-AN: 109307394
expires: Thu, 21 Sep 1995 13:12:40 GMT
approved: avg@cwi.nl (Annius Groenink)

sender: news@cwi.nl (The Daily Dross)
organization: CWI, Amsterdam
newsgroups: comp.sources.atari.st,comp.sys.atari.st
originator: avg@mijt.cwi.nl

Submitted-by: avg@cwi.nl (Annius Groenink)
Posting-number: Volume 8, Info 7

This article is posted twice a month. It gives a short introduction to comp.sources.atari.st and describes how to submit sources or digressions to the group.

I am always looking for suggestions on how to improve the usefulness of the newsgroup, and can be contacted at the following internet address:

avg@cwi.nl

Annius Groenink, Moderator.

ABOUT comp.sources.atari.st

The moderated usenet group comp.sources.atari.st contains publications or announcements of sources and texts about programming for Atari TOS computers. The publications on comp.sources.atari.st may be freely distributed as long as modifications are documented and the original authors are mentioned.

THIS IS A MODERATED GROUP. YOU CANNOT POST TO IT BY YOURSELF. Read the instructions below for submitting material intended for publication on comp.sources.atari.st.

SUBMITTING SOURCES OR DIGRESSIONS FOR PUBLICATION

Items intended for posting or queries and problem notes should be sent to avg@cwi.nl. If you are on a UUCP-only site, you can send them to uunet!cwi.nl!avg.

If you want verification of arrival, say so at the beginning of your submission, or in a cover note. I try to verify that a program works, and if I can't get it to work, I may hold up posting it for a couple of days. Please note that, except in rare cases, software without documentation will not be published. The backlog from receipt to posting varies from one to four weeks depending mostly on the set of submissions currently in my queue.

If you are submitting both sources and binaries, PLEASE send the two separately. If I have to separate your sources from your binaries by

hand, your submission will most likely sit on the back burner for a while, or be rejected outright.

Sources can be submitted in ASCII form, but for large pacages, a ZOO 2.1 archive encoded using Dumas uuencode is preferred.

Please include a description of your program at the top of your article. Look at old issues of comp.binaries.atari.st for examples of the sorts of descriptions I'm looking for. In general you should indicate what the program does, on what types of Atari TOS computers, whether your program requires a certain amount of on-board memory, a hard disk or a certain screen resolution. If necessary, I may rewrite or add to this information in order to make the postings to the group more uniform. I can't check your submission in all possible configurations, and see what works and what doesn't. Without a description, your submission is likely to be rejected.

THE STRUCTURE OF COMP.SOURCES.ATARI.ST ARTICLES

There are two types of articles in comp.binaries.atari.st: "information postings" and "sources". The source postings are also called "Issues". For a while, 'announcements' were also accepted, but there is now a dedicated group, comp.sys.atari.announce, for that purpose. The type of a posting can be distinguished by the subject line:

```
Subject: v02INF1: Introduction to comp.sources.atari.st
```

This first word in the title identifies this as the first info posting of volume two. Similarly, the subject line shown below:

```
Subject: v06i081: godsrc -- Universe simulator
```

identifies this as the 81st source article in Volume 6. Large sources are broken up into smaller pieces, and have subject lines that look like this:

```
Subject: v06i088: decbustr -- VMS clone in ST-LOGO, Part03/08
```

Certain information about the system configuration required to use the program is given on the keywords line.

```
Keywords: MWC, Megamax, 1meg, high
```

This means that the program compiles with Mark Williams or Megamax C, requires at least one meg of RAM and only runs in high resolution. Following is a list of keywords; new ones may be added as needed. They are mostly self- explanatory.

```
  Alcyon  - Alcyon C (included with Atari developer's kit)
  MWC         - Mark Williams C
  Megamax - Megamax C
  Laser       - Laser C
  Lattice - Lattice C
  C       - Any C compiler
  OSS         - OSS Personal Pascal
  BASIC       - ST BASIC (this is not a joke!)
  uuencode    - program is uuencoded (uudecode required to unpack)
  arc         - program is archived (arc required to unpack)
  zoo-high    - program is archived (Zoo 2.1 required to unpack)
  lharc       - program is archived (lharc required to unpack)
  high        - high resolution
  medium  - medium resolution
  low         - low resolution
  1meg        - needs 1 meg of RAM
```

The References: line, if present, points to the previous part of a multi-part program, so that newsreaders (trn, nn, etc.) which can follow threads will recognize a large program as a single thread.

The first few lines of an article are auxiliary headers that look like this:

```
  Submitted-by: david@bdt.UUCP (David Beckmeyer)
  Posting-number: Volume 6, Issue 80
  Archive-name: micro-rtx
```

The "Submitted by" is the author of the program. If you have comments about the sources published in comp.sources.atari.st this is the person to contact. When possible, this address is in domain form, otherwise it is a UUCP bang path relative to some major (backbone) site.

The second line repeats the volume/issue information for the aide of notes sites and automatic archiving programs.

The Archive-name is the "official" name of this source in the archive. Large postings will have names that look like this:

```
Archive-name: desktop/part01
```

REPORTING AND TRACKING BUGS AND PATCHES TO POSTINGS

Most of the time, bugs and updates will be posted to comp.sys.atari.st. Sometimes, "official" updates from the author will be posted submitted to comp.sources.atari.st so that they are archived with the sources.

Since there is no easy way to apply source code patches on the ST, changes to a program often result in a whole new version being posted.

To report bugs, contact the person listed in the Submitted-to header. Often there is a contact address in a README file, too. I do not maintain the sources I moderate, so don't send your bug reports to me.

HOW TO OBTAIN OLD ISSUES

There is currently no officially documented archive for this group.

--
Annius V. Groenink | avg@cwi.nl | Private/Edith/ZFC:
CWI, Kruislaan 413 | zfc@zfc.nl | P.O. Box 12079
1098 SJ Amsterdam | Room M233 ext. 4077 | NL 1100 AB Amsterdam
The Netherlands | Phone: +31 20 592 4077 | Phone: +31 20 695 9901

From 225317497161103182
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,8b553eecc734ec6d,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,8b553eecc734ec6d,start
X-Google-Attributes: gid100493,public
From: avg@cwi.nl (Annius Groenink)
Subject: v08INF8: Introduction to comp.sources.atari.st
Date: 1995/09/15
Message-ID: <DEy2DL.9ru@cwi.nl>
X-Deja-AN: 110190243
expires: Thu, 5 Oct 1995 11:24:57 GMT
approved: avg@cwi.nl (Annius Groenink)
sender: news@cwi.nl (The Daily Dross)
organization: CWI, Amsterdam
newsgroups: comp.sources.atari.st,comp.sys.atari.st
originator: avg@mijt.cwi.nl

Submitted-by: avg@cwi.nl (Annius Groenink)
Posting-number: Volume 8, Info 8

This article is posted twice a month. It gives a short introduction to comp.sources.atari.st and describes how to submit sources or digressions to the group.

I am always looking for suggestions on how to improve the usefulness of the newsgroup, and can be contacted at the following internet address:

avg@cwi.nl

Annius Groenink, Moderator.

ABOUT comp.sources.atari.st

The moderated usenet group comp.sources.atari.st contains publications or announcements of sources and texts about programming for Atari TOS computers. The publications on comp.sources.atari.st may be freely distributed as long as modifications are documented and the original authors are mentioned.

THIS IS A MODERATED GROUP. YOU CANNOT POST TO IT BY YOURSELF. Read the instructions below for submitting material intended for publication on comp.sources.atari.st.

SUBMITTING SOURCES OR DIGRESSIONS FOR PUBLICATION

Items intended for posting or queries and problem notes should be sent to avg@cwi.nl. If you are on a UUCP-only site, you can send them to uunet!cwi.nl!avg.

If you want verification of arrival, say so at the beginning of your submission, or in a cover note. I try to verify that a program works, and if I can't get it to work, I may hold up posting it for a couple of days. Please note that, except in rare cases, software without documentation will not be published. The backlog from receipt to posting varies from one to four weeks depending mostly on the set of submissions currently in my queue.

If you are submitting both sources and binaries, PLEASE send the two separately. If I have to separate your sources from your binaries by hand, your submission will most likely sit on the back burner for a while, or be rejected outright.

Sources can be submitted in ASCII form, but for large pacages, a ZOO 2.1 archive encoded using Dumas uuencode is preferred.

Please include a description of your program at the top of your article. Look at old issues of comp.binaries.atari.st for examples of the sorts of descriptions I'm looking for. In general you should indicate what the program does, on what types of Atari TOS computers, whether your program requires a certain amount of on-board memory, a hard disk or a certain screen resolution. If necessary, I may rewrite or add to this information in order to make the postings to the group more uniform. I can't check your submission in all possible configurations, and see what works and what doesn't. Without a description, your submission is likely to be rejected.

THE STRUCTURE OF COMP.SOURCES.ATARI.ST ARTICLES

There are two types of articles in comp.binaries.atari.st: "information postings" and "sources". The source postings are also called "Issues". For a while, 'announcements' were also accepted, but there is now a dedicated group, comp.sys.atari.announce, for that purpose. The type of a posting can be distinguished by the subject line:

```
Subject: v02INF1: Introduction to comp.sources.atari.st
```

This first word in the title identifies this as the first info posting of volume two. Similarly, the subject line shown below:

```
Subject: v06i081: godsrc -- Universe simulator
```

identifies this as the 81st source article in Volume 6. Large sources are broken up into smaller pieces, and have subject lines that look like this:

```
Subject: v06i088: decbustr -- VMS clone in ST-LOGO, Part03/08
```

Certain information about the system configuration required to use the program is given on the keywords line.

```
Keywords: MWC, Megamax, 1meg, high
```

This means that the program compiles with Mark Williams or Megamax C, requires at least one meg of RAM and only runs in high resolution. Following is a list of keywords; new ones may be added as needed. They are mostly self- explanatory.

```
Alcyon  - Alcyon C (included with Atari developer's kit)
MWC         - Mark Williams C
Megamax - Megamax C
Laser       - Laser C
Lattice - Lattice C
C       - Any C compiler
OSS         - OSS Personal Pascal
BASIC       - ST BASIC (this is not a joke!)
uuencode    - program is uuencoded (uudecode required to unpack)
arc         - program is archived (arc required to unpack)
zoo-high    - program is archived (Zoo 2.1 required to unpack)
lharc       - program is archived (lharc required to unpack)
high        - high resolution
medium  - medium resolution
low         - low resolution
1meg        - needs 1 meg of RAM
```

The References: line, if present, points to the previous part of a multi-part program, so that newsreaders (trn, nn, etc.) which can follow threads will recognize a large program as a single thread.

The first few lines of an article are auxiliary headers that look like this:

```
Submitted-by: david@bdt.UUCP (David Beckmeyer)
Posting-number: Volume 6, Issue 80
Archive-name: micro-rtx
```

The "Submitted by" is the author of the program. If you have comments about the sources published in comp.sources.atari.st this is the person to contact. When possible, this address is in domain form, otherwise it is a UUCP bang path relative to some major (backbone) site.

The second line repeats the volume/issue information for the aide of notes sites and automatic archiving programs.

The Archive-name is the "official" name of this source in the archive. Large postings will have names that look like this:

Archive-name: desktop/part01

REPORTING AND TRACKING BUGS AND PATCHES TO POSTINGS

Most of the time, bugs and updates will be posted to comp.sys.atari.st. Sometimes, "official" updates from the author will be posted submitted to comp.sources.atari.st so that they are archived with the sources.

Since there is no easy way to apply source code patches on the ST, changes to a program often result in a whole new version being posted.

To report bugs, contact the person listed in the Submitted-to header. Often there is a contact address in a README file, too. I do not maintain the sources I moderate, so don't send your bug reports to me.

HOW TO OBTAIN OLD ISSUES

There is currently no officially documented archive for this group.

--
Annius V. Groenink | avg@cwi.nl | Private/Edith/ZFC:
CWI, Kruislaan 413 | zfc@zfc.nl | P.O. Box 12079
1098 SJ Amsterdam | Room M233 ext. 4077 | NL 1100 AB Amsterdam
The Netherlands | Phone: +31 20 592 4077 | Phone: +31 20 695 9901

From -2599364866568628953
X-Google-Language: ENGLISH,ASCII-7-bit

X-Google-Thread: 100493,f573113997c00bec,start
X-Google-Attributes: gid100493,public
From: avg@cwi.nl (Annius Groenink)
Subject: Discussion on the future of this group
Date: 1996/02/15
Message-ID: <avg.824400294@news.cwi.nl>#1/1
X-Deja-AN: 139595860
sender: news@cwi.nl (The Daily Dross)
organization: CWI, Amsterdam
newsgroups: comp.sources.atari.st

This is an informal `request for discussion' on the moderated Atari news groups

   comp.binaries.atari.st
   comp.sys.atari.st

A moderated news group, as opposed to a regular one such as comp.sys.atari.st, is a news group people can not directly post to---instead a human `moderator' receives all posted articles, and checks their content before they are actually posted.

The Atari binaries/sources news groups were created in times that ftp was not yet very popular, and the world wide web didn't exist; the news was a good way of distributing new software.

The groups are hardly used anymore. Most people prefer to distribute their software through ftp or through their home pages, where they can add as much information as they want, in a pleasant form of presentation.

Here's an indication of what is submitted these days on a MONTHLY basis:

- 2 regular postings at most

- 20 posts saying `what are these groups for?' `where is everybody'?

- 30 advertisements `who wants to buy my racy 520st?'

- 60 spams `100 sexy girls ready at there phones craving for your hard cock'

I therefore propose to permanently delete the groups from usenet. I don't think it is wise to continue the groups as they are now, and frankly not many people will miss them. Instead, one could make their programs available by ftp and post an announcement on one of the other Atari groups, or perhaps, if it is just a small source file, post it directly to comp.sys.atari.st.

I have checked with the authorities, and it doesn't seem necessary for such a low level group to go through a formal voting system. If someone

were to take over the moderation job, they would experience they same problems as I am suffering now, and there simply is no way of improving the situation, nor is there any real need to do so.

Please note that I am not saying that I am looking for a new moderator, but would like to discuss the usefulness of the groups in general.

So, if there are no serious objections, I will request deletion of the groups somewhere early March.

--
Annius V. Groenink ..... avg@cwi.nl
Room M233, CWI, Kruislaan 413, 1098 SJ Amsterdam phone +31 20 592 40 77
.. and ZFC/home P.O.Box 12079, 1100 AB Amsterdam phone +31 20 4 208 248
.. zfc@zfc.nl .....

From 8221227371078964267
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,62cf0405e993a2d8,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,62cf0405e993a2d8,start
X-Google-Attributes: gid100493,public
X-Google-Thread: fa0ea,62cf0405e993a2d8,start
X-Google-Attributes: gidfa0ea,public
X-Google-Thread: 15ae9a,62cf0405e993a2d8,start
X-Google-Attributes: gid15ae9a,public
X-Google-Thread: 1055eb,62cf0405e993a2d8,start
X-Google-Attributes: gid1055eb,public
X-Google-Thread: 156df7,62cf0405e993a2d8,start
X-Google-Attributes: gid156df7,public
X-Google-Thread: 1049df,689dcb0ea640793d
X-Google-Attributes: gid1049df,public
X-Google-Thread: 107ef2,62cf0405e993a2d8,start
X-Google-Attributes: gid107ef2,public
From: Ulfert Kr?ger <ulfert.krueger@elmshorn.netsurf.de>
Subject: Re: Atari software
Date: 1996/12/31
Message-ID: <32C85E6B.6811@elmshorn.netsurf.de>#1/1
X-Deja-AN: 206860284
references: <01bbe4b6$68e46540$LocalHost@mickpich>
<01bbed36$8eeaf3c0$640847cc@office8.csra.net>
content-type: text/plain; charset=us-ascii
organization: shlink.de
mime-version: 1.0
reply-to: ulfert.krueger@elmshorn.netsurf.de
newsgroups:
alt.atari,alt.binaries.atari,alt.crews.atari,comp.binaries.atari.st,comp.sources.atari.st,comp.sys.atari.st,ctdl.sys.atari.st,maus.sys.atari.software,rec.games.video.atari,stgt.atari
x-mailer: Mozilla 3.0 (Win95; I)

Greg N. Weinstein wrote:
>
> Mickael Pichot <mickael.pichot@skynet.be> wrote in article
> <01bbe4b6$68e46540$LocalHost@mickpich>...
> > Where can I find softwares for my atari emulator?
> >
> > Mickael.
> >
>   Try ftp://ftp3.cyberspy.com/atari8/yogis_site/BINARIES/IMAGES/ if you're
> looking for games. BUT, as I posted elsewhere on this group, the files you
> unzip are in ATR format. I don't yet know how to use these files, but they
> have some very good software, especially games001.zip and games002.zip.
> Good luck!

Try to rename to *.prg or *.tos (the usualy atari suffixes).
Maybe it works.
Is the emulator a software or a hardware release ?

From -6709413928903379515
X-Google-Language: ENGLISH,ASCII
X-Google-Thread: 111a55,62cf0405e993a2d8
X-Google-Attributes: gid111a55,public
X-Google-Thread: 1055eb,62cf0405e993a2d8
X-Google-Attributes: gid1055eb,public
X-Google-Thread: 100493,62cf0405e993a2d8
X-Google-Attributes: gid100493,public
X-Google-Thread: 1049df,689dcb0ea640793d
X-Google-Attributes: gid1049df,public
X-Google-Thread: 15ae9a,62cf0405e993a2d8
X-Google-Attributes: gid15ae9a,public
X-Google-Thread: 107ef2,62cf0405e993a2d8
X-Google-Attributes: gid107ef2,public
X-Google-Thread: fa0ea,62cf0405e993a2d8
X-Google-Attributes: gidfa0ea,public
X-Google-Thread: 156df7,62cf0405e993a2d8
X-Google-Attributes: gid156df7,public
From: "Rory McMahon" <mcmahon@inil.com>
Subject: Re: Atari software
Date: 1996/12/31
Message-ID: <01bbf706$3b102980$a9ab1fce@mcmahon>#1/1
X-Deja-AN: 206917588
references: <01bbe4b6$68e46540$LocalHost@mickpich>
<01bbed36$8eeaf3c0$640847cc@office8.csra.net> <32C85E6B.6811@elmshorn.netsurf.de>
newsgroups:
alt.atari,alt.binaries.atari,alt.crews.atari,comp.binaries.atari.st,comp.sources.a
tari.st,comp.sys.atari.st,ctdl.sys.atari.st,maus.sys.atari.software,rec.games.vide
o.atari,stgt.atari

Ulfert Kr@ger <ulfert.krueger@elmshorn.netsurf.de> wrote in article
<32C85E6B.6811@elmshorn.netsurf.de>...
> Greg N. Weinstein wrote:
> >
> > Mickael Pichot <mickael.pichot@skynet.be> wrote in article
> > <01bbe4b6$68e46540$LocalHost@mickpich>...
> > > Where can I find softwares for my atari emulator?
> > >
> > > Mickael.
> > >
> >   Try ftp://ftp3.cyberspy.com/atari8/yogis_site/BINARIES/IMAGES/ if
you're
> > looking for games. BUT, as I posted elsewhere on this group, the files
you
> > unzip are in ATR format. I don't yet know how to use these files, but
they
> > have some very good software, especially games001.zip and games002.zip.
> > Good luck!
>
> Try to rename to *.prg or *.tos (the usualy atari suffixes).
> Maybe it works.
> Is the emulator a software or a hardware release ?
>

The zip files are regular PC or ST zip files... you would need STZip or
PKUNZIP to extract the files inside...

From 4964950722359565770
X-Google-Language: ENGLISH,ASCII
X-Google-Thread: 111a55,62cf0405e993a2d8
X-Google-Attributes: gid111a55,public
X-Google-Thread: 1055eb,62cf0405e993a2d8
X-Google-Attributes: gid1055eb,public
X-Google-Thread: fa0ea,62cf0405e993a2d8
X-Google-Attributes: gidfa0ea,public
X-Google-Thread: 1049df,689dcb0ea640793d
X-Google-Attributes: gid1049df,public
X-Google-Thread: 15ae9a,62cf0405e993a2d8
X-Google-Attributes: gid15ae9a,public
X-Google-Thread: 100493,62cf0405e993a2d8
X-Google-Attributes: gid100493,public
X-Google-Thread: 156df7,62cf0405e993a2d8
X-Google-Attributes: gid156df7,public
X-Google-Thread: 107ef2,62cf0405e993a2d8
X-Google-Attributes: gid107ef2,public
From: infoman@xmission.xmission.com (InfoMan)
Subject: Re: Atari software
Date: 1996/12/31
Message-ID: <5abmem$qt0@xmission.xmission.com>#1/1

X-Deja-AN: 206978134
references: <01bbe4b6$68e46540$LocalHost@mickpich>
<01bbed36$8eeaf3c0$640847cc@office8.csra.net> <32C85E6B.6811@elmshorn.netsurf.de>
content-type: text/plain; charset=US-ASCII
organization: XMission Internet (801 539 0900)
mime-version: 1.0
newsgroups:
alt.atari,alt.binaries.atari,alt.crews.atari,comp.binaries.atari.st,comp.sources.atari.st,comp.sys.atari.st,ctdl.sys.atari.st,maus.sys.atari.software,rec.games.video.atari,stgt.atari

In alt.atari Ulfert Kr[illegible]ger <ulfert.krueger@elmshorn.netsurf.de> wrote:
: Greg N. Weinstein wrote:
: >
: > Mickael Pichot <mickael.pichot@skynet.be> wrote in article
: > <01bbe4b6$68e46540$LocalHost@mickpich>...
: > > Where can I find softwares for my atari emulator?
: > >
: > > Mickael.
: > >
: >   Try ftp://ftp3.cyberspy.com/atari8/yogis_site/BINARIES/IMAGES/ if you're
: > looking for games. BUT, as I posted elsewhere on this group, the files you
: > unzip are in ATR format. I don't yet know how to use these files, but they
: > have some very good software, especially games001.zip and games002.zip.
: > Good luck!

: Try to rename to *.prg or *.tos (the usualy atari suffixes).
: Maybe it works.
: Is the emulator a software or a hardware release ?

The extenders .prg and .tos are NOT for the 8-bit. Most emulators will
take .atr files, or it is very easy to convert to the other formats.
I have some software (emulators, etc.) available on my Atari site, along
with FAQs and other info on how to use them:
http://infoman.simplenet.com/atari/

/InfoMan

From -185231770655534775
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,62cf0405e993a2d8
X-Google-Attributes: gid111a55,public
X-Google-Thread: 156df7,62cf0405e993a2d8
X-Google-Attributes: gid156df7,public
X-Google-Thread: 100493,62cf0405e993a2d8
X-Google-Attributes: gid100493,public
X-Google-Thread: 107ef2,62cf0405e993a2d8
X-Google-Attributes: gid107ef2,public
X-Google-Thread: 1055eb,62cf0405e993a2d8

X-Google-Attributes: gid1055eb,public
X-Google-Thread: fa0ea,62cf0405e993a2d8
X-Google-Attributes: gidfa0ea,public
X-Google-Thread: 1049df,689dcb0ea640793d
X-Google-Attributes: gid1049df,public
X-Google-Thread: 15ae9a,62cf0405e993a2d8
X-Google-Attributes: gid15ae9a,public
From: Mykle <mykle@con2.com>
Subject: Atari Web Browser
Date: 1997/01/01
Message-ID: <32CA6724.12CE@con2.com>#1/1
X-Deja-AN: 207107148
references: <01bbe4b6$68e46540$LocalHost@mickpich>
<01bbed36$8eeaf3c0$640847cc@office8.csra.net> <32C85E6B.6811@elmshorn.netsurf.de>
<01bbf706$3b102980$a9ab1fce@mcmahon>
content-type: text/plain; charset=us-ascii
organization: Connect2 InterNet News Site
mime-version: 1.0
reply-to: mykle@con2.com
newsgroups:
alt.atari,alt.binaries.atari,alt.crews.atari,comp.binaries.atari.st,comp.sources.a
tari.st,comp.sys.atari.st,ctdl.sys.atari.st,maus.sys.atari.software,rec.games.vide
o.atari,stgt.atari
x-mailer: Mozilla 3.0 (Macintosh; I; 68K)

hi,

Is there an Atari Web Browser? If so, will it run on an Atari 1040Stf?
TOS 1.0? Where can I get my hands on it? Is it shareware? Commercial?
Please e-mail me with info as I don't get all the newsgroups.

Thanks

Mykle

From 79589233026695071
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,62cf0405e993a2d8
X-Google-Attributes: gid111a55,public
X-Google-Thread: 1055eb,62cf0405e993a2d8
X-Google-Attributes: gid1055eb,public
X-Google-Thread: fa0ea,62cf0405e993a2d8
X-Google-Attributes: gidfa0ea,public
X-Google-Thread: 15ae9a,62cf0405e993a2d8
X-Google-Attributes: gid15ae9a,public
X-Google-Thread: 107ef2,62cf0405e993a2d8
X-Google-Attributes: gid107ef2,public
X-Google-Thread: 156df7,62cf0405e993a2d8
X-Google-Attributes: gid156df7,public
X-Google-Thread: 1049df,689dcb0ea640793d

X-Google-Attributes: gid1049df,public
X-Google-Thread: 100493,62cf0405e993a2d8
X-Google-Attributes: gid100493,public
From: Mykle <mykle@con2.com>
Subject: Sell me your Atari junk!
Date: 1997/01/01
Message-ID: <32CA6C4B.5A42@con2.com>#1/1
X-Deja-AN: 207109678
references: <01bbe4b6$68e46540$LocalHost@mickpich>
<01bbed36$8eeaf3c0$640847cc@office8.csra.net> <32C85E6B.6811@elmshorn.netsurf.de>
<01bbf706$3b102980$a9ab1fce@mcmahon>
content-type: text/plain; charset=us-ascii
organization: Connect2 InterNet News Site
mime-version: 1.0
reply-to: mykle@con2.com
newsgroups:
alt.atari,alt.binaries.atari,alt.crews.atari,comp.binaries.atari.st,comp.sources.a
tari.st,comp.sys.atari.st,ctdl.sys.atari.st,maus.sys.atari.software,rec.games.vide
o.atari,stgt.atari
x-mailer: Mozilla 3.0 (Macintosh; I; 68K)

Hi,

Keep those offers coming!

Some of you know me. I've bought quite a bit of atari "junk" in the past
two weeks, from complete ST systems to hardware to software. Well, I'm
not done yet. I still need software and I still need an external floppy
for an Atari 1040Stf. I "WILL" buy from the best offers that I receive
so send me your lists and get some money for that junk and regain some
closet space.

Hurry, I won't be buying forever and you'll be stuck with it!

Mykle@con2.com

From -6757872802128053173
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,62cf0405e993a2d8
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,62cf0405e993a2d8
X-Google-Attributes: gid100493,public
X-Google-Thread: 1049df,689dcb0ea640793d
X-Google-Attributes: gid1049df,public
X-Google-Thread: 1055eb,62cf0405e993a2d8
X-Google-Attributes: gid1055eb,public
X-Google-Thread: fa0ea,62cf0405e993a2d8
X-Google-Attributes: gidfa0ea,public
X-Google-Thread: 107ef2,62cf0405e993a2d8
X-Google-Attributes: gid107ef2,public

X-Google-Thread: 15ae9a,62cf0405e993a2d8
X-Google-Attributes: gid15ae9a,public
X-Google-Thread: 156df7,62cf0405e993a2d8
X-Google-Attributes: gid156df7,public
From: atari@teleport.com (Portland Atari Club)
Subject: Re: Atari software
Date: 1997/01/01
Message-ID: <5aeec5$3jn$3@nadine.teleport.com>#1/1
X-Deja-AN: 207226148
references: <01bbe4b6$68e46540$LocalHost@mickpich>
<01bbed36$8eeaf3c0$640847cc@office8.csra.net> <32C85E6B.6811@elmshorn.netsurf.de>
<5abmem$qt0@xmission.xmission.com>
followup-to:
alt.atari,alt.binaries.atari,alt.crews.atari,comp.binaries.atari.st,comp.sources.a
tari.st,comp.sys.atari.st,ctdl.sys.atari.st,maus.sys.atari.software,rec.games.vide
o.atari,stgt.atari
organization: Teleport - Portland's Public Access (503) 220-1016
newsgroups:
alt.atari,alt.binaries.atari,alt.crews.atari,comp.binaries.atari.st,comp.sources.a
tari.st,comp.sys.atari.st,ctdl.sys.atari.st,maus.sys.atari.software,rec.games.vide
o.atari,stgt.atari

This discussion does not belong in alt.binaries.atari - you can discuss it
in alt.binaries.atari.d if you like, but future articles in this thread,
in a.b.a, will be cancelled.

Please read the newsgroup charter.

--
----------------------http://www.teleport.com/~atari----------------------
  ATARI COMPUTING - the new 60 PAGE printed magazine for all Atari users
     NEWS - GOSSIP - FEATURES - REVIEWS - HUMOUR - HOW TO DO IT - Q&A
          email for subscription details and further information
  Europe: mkerslake@cix.compulink.co.uk   USA: atari@teleport.com
--------------------ftp://ftp.teleport.com/users/atari--------------------

From -3861442149543013367
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,62cf0405e993a2d8
X-Google-Attributes: gid111a55,public
X-Google-Thread: 107ef2,62cf0405e993a2d8
X-Google-Attributes: gid107ef2,public
X-Google-Thread: 1049df,689dcb0ea640793d
X-Google-Attributes: gid1049df,public
X-Google-Thread: 15ae9a,62cf0405e993a2d8
X-Google-Attributes: gid15ae9a,public
X-Google-Thread: 100493,62cf0405e993a2d8

X-Google-Attributes: gid100493,public
X-Google-Thread: 156df7,62cf0405e993a2d8
X-Google-Attributes: gid156df7,public
X-Google-Thread: fa0ea,62cf0405e993a2d8
X-Google-Attributes: gidfa0ea,public
X-Google-Thread: 1055eb,62cf0405e993a2d8
X-Google-Attributes: gid1055eb,public
From: ntw5miku@rhds01.rz.fht-esslingen.de (Michael Kugelmann)
Subject: Re: Atari Web Browser
Date: 1997/01/01
Message-ID: <32cc2f1c1a01c70@rhds01.rz.fht-esslingen.de>#1/1
X-Deja-AN: 207359172
references: <32CA6724.12CE@con2.com>
x-hardware: 68000/8 Atari 1040STF (connected with SOUP)
organization: Fachhochschule fuer Technik, Esslingen
reply-to: ntw5miku@rz.fht-esslingen.de
newsgroups:
alt.atari,alt.binaries.atari,alt.crews.atari,comp.binaries.atari.st,comp.sources.atari.st,comp.sys.atari.st,ctdl.sys.atari.st,maus.sys.atari.software,rec.games.video.atari,stgt.atari
originator: ntw5miku@rhds23

Mykle schrieb:

M>Is there an Atari Web Browser?

yes, there are two:
1) CAB (from Alexander Clauss, Germany) fine working, together with STIK
2) ??? another one, who just can only read local files and who ist just in a
   early development state.

M>If so, will it run on an Atari 1040Stf?

yes, but 1MByte of RAM is very less!

M>TOS 1.0?

??? i'm not shure... TOS 1.2 (I guess) or better 1.4 (shure) ist preferred.

M>Where can I get my hands on it?

Somewhere in Germany, visit the STIK support page, there is a link.
Or use archie for seraching.

Michael.

From 3663011179125875552

X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,62cf0405e993a2d8
X-Google-Attributes: gid111a55,public
X-Google-Thread: 1055eb,62cf0405e993a2d8
X-Google-Attributes: gid1055eb,public
X-Google-Thread: fa0ea,62cf0405e993a2d8
X-Google-Attributes: gidfa0ea,public
X-Google-Thread: 100493,62cf0405e993a2d8
X-Google-Attributes: gid100493,public
X-Google-Thread: 156df7,62cf0405e993a2d8
X-Google-Attributes: gid156df7,public
X-Google-Thread: 1049df,689dcb0ea640793d
X-Google-Attributes: gid1049df,public
X-Google-Thread: 15ae9a,62cf0405e993a2d8
X-Google-Attributes: gid15ae9a,public
X-Google-Thread: 107ef2,62cf0405e993a2d8
X-Google-Attributes: gid107ef2,public
From: gregor@goldengate.net
Subject: Re: Atari Web Browser
Date: 1997/01/03
Message-ID: <32cc760a.10870431@news.goldengate.net>#1/1
X-Deja-AN: 207391249
references: <32CA6724.12CE@con2.com> <32cc2f1c1a01c70@rhds01.rz.fht-esslingen.de>
newsgroups:
alt.atari,alt.binaries.atari,alt.crews.atari,comp.binaries.atari.st,comp.sources.atari.st,comp.sys.atari.st,ctdl.sys.atari.st,maus.sys.atari.software,rec.games.video.atari,stgt.atari

On Wed, 01 Jan 1997 19:08:50 +0100, ntw5miku@rhds01.rz.fht-esslingen.de (Michael Kugelmann) wrote:

>Mykle schrieb:
>
>M>Is there an Atari Web Browser?
>
>yes, there are two:
>1) CAB (from Alexander Clauss, Germany) fine working, together with STIK

>Or use archie for seraching.

A simple way to find it is by visiting the Hensa Archive. they always have the latest version of the program.
The address is not at hand because I've become a platform traitor--the PC is handier but will never replace the Mega for
MIDI.
Resend your question to comp.sys.atari.st and ask for the address. The Hensa archive is pretty comprehensive for Atari
SW and apps. The Usenet group is very very helpful for question and direction.

****
* Bradley F. Maurer *

* gregor@goldengate.net   *
* Minneapolis, Minnesota USA   *
* http://www.goldengate.net/~gregor *
****

From -2788107954320999009
X-Google-Language: ENGLISH,ASCII
X-Google-Thread: 111a55,62cf0405e993a2d8
X-Google-Attributes: gid111a55,public
X-Google-Thread: 1049df,689dcb0ea640793d
X-Google-Attributes: gid1049df,public
X-Google-Thread: 100493,62cf0405e993a2d8
X-Google-Attributes: gid100493,public
X-Google-Thread: 15ae9a,62cf0405e993a2d8
X-Google-Attributes: gid15ae9a,public
X-Google-Thread: 107ef2,62cf0405e993a2d8
X-Google-Attributes: gid107ef2,public
X-Google-Thread: 156df7,62cf0405e993a2d8
X-Google-Attributes: gid156df7,public
X-Google-Thread: fa0ea,62cf0405e993a2d8
X-Google-Attributes: gidfa0ea,public
X-Google-Thread: 1055eb,62cf0405e993a2d8
X-Google-Attributes: gid1055eb,public
From: rbennett@jagunet.com (Roger R. Bennett)
Subject: Re: Atari software
Date: 1997/01/04
Message-ID: <5akdto$jmq@skydiver.jaguNET.com>#1/1
X-Deja-AN: 207593476
references: <01bbe4b6$68e46540$LocalHost@mickpich>
<01bbed36$8eeaf3c0$640847cc@office8.csra.net> <32C85E6B.6811@elmshorn.netsurf.de>
organization: jaguNET Access Services
newsgroups:
alt.atari,alt.binaries.atari,alt.crews.atari,comp.binaries.atari.st,comp.sources.a
tari.st,comp.sys.atari.st,ctdl.sys.atari.st,maus.sys.atari.software,rec.games.vide
o.atari,stgt.atari

Ulfert Kr�ger <ulfert.krueger@elmshorn.netsurf.de> wrote:

>Greg N. Weinstein wrote:
>>
>> Mickael Pichot <mickael.pichot@skynet.be> wrote in article
>> <01bbe4b6$68e46540$LocalHost@mickpich>...
>> > Where can I find softwares for my atari emulator?
>> >
>> > Mickael.
>> >
>>   Try ftp://ftp3.cyberspy.com/atari8/yogis_site/BINARIES/IMAGES/ if you're
>> looking for games. BUT, as I posted elsewhere on this group, the files you
>> unzip are in ATR format. I don't yet know how to use these files, but they

>> have some very good software, especially games001.zip and games002.zip.
>> Good luck!

>Try to rename to *.prg or *.tos (the usualy atari suffixes).
>Maybe it works.
>Is the emulator a software or a hardware release ?

The .atr files are disk images of old Atari stuff. They are used by
Atari 8-bit emulators, of which I recommend XL-it - it's the best
shareware emulator out there. It's available by link at Yogi's
Jellystone Park at
http:// www.blast.net/~scharles/

Good luck, and have fun!

R.R. Bennett (rbennett@jaguNET.com)
Editor, Dragon's Lair Webzine
http://www.jaguNET.com/~rbennett/dragon.htm
Science Fiction, Fantasy, and Horror

From 5329086613649927856
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,62cf0405e993a2d8
X-Google-Attributes: gid111a55,public
X-Google-Thread: 1055eb,62cf0405e993a2d8
X-Google-Attributes: gid1055eb,public
X-Google-Thread: 15ae9a,62cf0405e993a2d8
X-Google-Attributes: gid15ae9a,public
X-Google-Thread: 156df7,62cf0405e993a2d8
X-Google-Attributes: gid156df7,public
X-Google-Thread: fa0ea,62cf0405e993a2d8
X-Google-Attributes: gidfa0ea,public
X-Google-Thread: 1049df,689dcb0ea640793d
X-Google-Attributes: gid1049df,public
X-Google-Thread: 107ef2,62cf0405e993a2d8
X-Google-Attributes: gid107ef2,public
X-Google-Thread: 100493,62cf0405e993a2d8
X-Google-Attributes: gid100493,public
From: atari@teleport.com.REMOVE.TO.REPLY (Rich Tietjens)
Subject: Cross-posts to alt.binaries.atari
Date: 1997/01/04
Message-ID: <5al22njvp1@nadine.teleport.com>#1/1
X-Deja-AN: 207637893
references: <32CA6724.12CE@con2.com>
organization: Portland Atari Club
newsgroups:
alt.atari,alt.binaries.atari,alt.crews.atari,comp.binaries.atari.st,comp.sources.a
tari.st,comp.sys.atari.st,ctdl.sys.atari.st,maus.sys.atari.software,rec.games.vide
o.atari,stgt.atari

Recently, there has been a rash of cross-posted discussion articles in alt.binaries.atari. I responded (as moderator) by gently reminding everyone that the newsgroup is NOT for discussion, but only for binaries.

In response, I received rude email from a maus.net user (or possibly 2). Thus, the following new policy is in effect immediately:

Henceforth, ANY AND ALL discussion articles cross-posted to a.b.a will be summarily cancelled. Charter violations will be reported to the appropriate system adminstrators.

I am deliberately cross-posting this warning to all the previously-involved newsgroups. If people would simply read the charter, it wouldn't have been necessary to intrude; to those of you in the majority - that is, courteous and literate users - I apologize. To the one or two of you (who know who you are, at fr.maus.de) who are rude and arrogant, raspberries.

----------http://www.teleport.com/~atari----------
  ATARI COMPUTING - the new 60 PAGE printed magazine for all Atari users
     NEWS - GOSSIP - FEATURES - REVIEWS - HUMOUR - HOW TO DO IT - Q&A
          email for subscription details and further information
 Europe: ataricomputing@cix.compulink.co.uk   USA: atari@teleport.com
----------ftp://ftp.teleport.com/users/atari----------

From 3951972767570742542
X-Google-Language: GERMAN,ASCII-7-bit
X-Google-Thread: 111a55,2fa150d52e73c19,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 13a459,2fa150d52e73c19,start
X-Google-Attributes: gid13a459,public
X-Google-Thread: 10ded6,2fa150d52e73c19,start
X-Google-Attributes: gid10ded6,public
X-Google-Thread: 1049df,2fa150d52e73c19,start
X-Google-Attributes: gid1049df,public
X-Google-Thread: 100493,2fa150d52e73c19,start
X-Google-Attributes: gid100493,public

X-Google-Thread: 153a4a,2fa150d52e73c19,start
X-Google-Attributes: gid153a4a,public
X-Google-Thread: 1113fd,2fa150d52e73c19,start
X-Google-Attributes: gid1113fd,public
X-Google-Thread: 12f90e,2fa150d52e73c19,start
X-Google-Attributes: gid12f90e,public
From: sysop@amigo.ping.de (Armin Goehrke)
Subject: ZConnect-Modul
Date: 1997/08/10
Message-ID: <zp42a6aFSp0VZ23@fc-kid.amigo.ping.de>#1/1
X-Deja-AN: 265435945
Distribution: world
Summary: Suche ZCONNECT Software fuer ALPHABOX
Organization: AMIGO Box Dortmund
X-Gateway: ZCONNECT UR hit.handshake.de [DUUCP BETA vom 14.07.1997]
Reply-To: sysop@amigo.ping.de
X-ZC-Telefon: (Data) 0231-831983|817543|831984 (Fax) 0231-817543 (VOICE)
0231-831989
Newsgroups: t-netz.rechner.atari.8bit,t-netz.rechner.atari.diskussion,z-
netz.alt.rechner.atari.8bit,z-netz.alt.sauger-
info.atari,maus.sys.atari.software,maus.sys.atari.talk,comp.sources.atari.st,comp.
sys.atari.st,t-netz.rechner.atari,lifestyle.atari-
allgemein,ping.support.atari,maus.sys.atari,alt.atari,ger-
net.gernet.atari,comp.sources.atari,comp.sys.atari,comp.sys.atari.programmer,ger.a
tari,lightspeed.rechner.atari,lightspeed.rechner.atari.allgemein,lightspeed.rechne
r.atari.software

Ich bin auf der Suche nach einem ZCONNECT Modul fuer das Boxprogramm
ALPHABOX.
Gibt es das ? Wenn ja wo kann man es beziehen ?

Antworten bitte NUR in meine PM da ich die Bretter nicht beziehe und
ich nicht fuer mich suche.

i`m searching for the zconnect software for the alphabox bbs software.
anybody knows where can ich get it ?

From -5620161356389875498
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,df24d8f4722b833f,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 108ff0,df24d8f4722b833f,start
X-Google-Attributes: gid108ff0,public
X-Google-Thread: 100493,df24d8f4722b833f,start
X-Google-Attributes: gid100493,public
X-Google-Thread: 109a67,df24d8f4722b833f,start
X-Google-Attributes: gid109a67,public
From: fragmentk@labyrinth.net.au

Subject: using system exclusive on atari 1040st
Date: 1998/01/13
Message-ID: <34bb4d2f.3385040@news.labyrinth.net.au>#1/1
X-Deja-AN: 315510192
Organization: Labyrinth Connections Pty Ltd
Newsgroups:
comp.sources.atari.st,comp.sys.atari.st.tech,comp.sys.atari.st,alt.music.makers.el
ectronic

Could someone please help me.... i am trying to get system exclusive
to work when sending messages from my atari 1040 st (using Notator
3.0) to a midi unit.
please reply to newsgroups only..... friends account.
thanks all

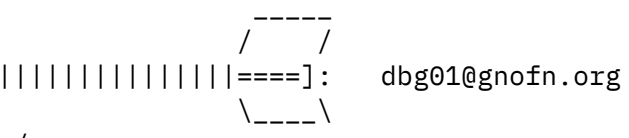
From -7425339745771575900
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,df24d8f4722b833f
X-Google-Attributes: gid111a55,public
X-Google-Thread: 108ff0,df24d8f4722b833f
X-Google-Attributes: gid108ff0,public
X-Google-Thread: 100493,df24d8f4722b833f
X-Google-Attributes: gid100493,public
X-Google-Thread: 109a67,df24d8f4722b833f
X-Google-Attributes: gid109a67,public
From: Dave Gostl <dbg01@gnofn.org>
Subject: Re: using system exclusive on atari 1040st
Date: 1998/01/13
Message-ID: <Pine.GSO.3.95.980113060657.27584A-100000@sparkie.gnofn.org>#1/1
X-Deja-AN: 315522363
References: <34bb4d2f.3385040@news.labyrinth.net.au>
Mime-Version: 1.0
To: fragmentk@labyrinth.net.au
Content-Type: TEXT/PLAIN; charset=US-ASCII
Organization: Greater New Orleans Free-Net
Newsgroups:
comp.sources.atari.st,comp.sys.atari.st.tech,comp.sys.atari.st,alt.music.makers.el
ectronic

On Tue, 13 Jan 1998 fragmentk@labyrinth.net.au wrote:

> Could someone please help me.... i am trying to get system exclusive
> to work when sending messages from my atari 1040 st (using Notator
> 3.0) to a midi unit.

Hey fragmentk!?! :-) What midi unit? Some midi modules (units?) just
accept sysex as is. Others want to "handshake' with other modules or
computers. My Proteus/1 works with any and all midi programs. But my
Ensoniq SQ-80 needs handshaking and other stuff.

Happy Festivus,
Dave
<pre>



</pre>
Bass player for Broadcast,
http://www.sstar.com/d
      on the web

From 4988482525440798276
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 1115ce,e0adbd549726a1bb,start
X-Google-Attributes: gid1115ce,public
X-Google-Thread: 100493,e0adbd549726a1bb,start
X-Google-Attributes: gid100493,public
From: strads@tmisnet.com (George Crissman)
Subject: Press Any Key....
Date: 1998/03/12
Message-ID: <35074ac0.410905@news.tmisnet.com>#1/1
X-Deja-AN: 333193381
Distribution: world
Organization: MaximumAccess Internet Services
Newsgroups: comp.sys.atari.programmer,comp.sources.atari.st

Greetings!

There was a recently-posted request that
programs should stop after printing text
to the screen (when exiting) so the user
could view the information before the
desktop reappears.

After some toying around with a variety of
commands, I've come up with a fairly-simple
(although heavily commented!) demonstration
function called pause_here(). It won't stop
existing programs, but programmers developing
new software will find this code easy to
implement.

It recognizes any user's keypress (even the
wiseguys who press SHIFT or CONTROL) with the
exception of CAPS LOCK, which apparently only
connects to the IKBD processor and not to the
Central Processing Unit. As a bonus, it will

recognize mouse clicks, as well.

A compiled GEM program containing this
function will be posted as "Two Bit Tutorial
Number 1" (probably in the binaries group)
soon. It will NOT need a resource (*.RSC)
file for proper operation, so don't worry
that it doesn't have one! (You can rename
it PAUSE.TOS, if the lack of a resource
file causes problems.)

-- George Crissman
-- strads@tmisnet.com
-- March 11, 1998

Internet Data Mining -- The Perfect Way To Dig For Data!
http://www.tmisnet.com/~strads/index.html

-- Source Code From 2B_TUT01.ZIP --

```
/***************** pause_here *****************/
/* Waits For Any Keypress (Except CAPS LOCK), */
/* Or Any Mouse Button Press.  Uses GEM Text  */
/* Mode, Typically Available During System    */
/* Boot And Program Termination.              */
/*                                            */
/* Bconstat() and Bconin() are capable of     */
/* recognizing ALL keypresses except for      */
/* CAPS LOCK.  Cconis(), Cconin(), Cconrs(),  */
/* and evnt_keybd() ignore a variety of keys  *
/* such as:  F1-F10, RETURN, ENTER, HELP,     */
/* UNDO, arrow keys, DELETE, INSERT, and      */
/* CLRHOME.                                   */
/*                                            */
/* Note -- To enhance readability:  explicit  */
/* comparisons are shown ( x == y ), "if"     */
/* blocks are surrounded by { } even though   */
/* not required for single statements, all    */
/* fixed numbers have been #defined, and      */
/* there are absolutely way too many comments.*/
/*                                            */
/* George Crissman              March 11, 1998 */
/* C-Lab Falcon MKII  GEM/TOS   Lattice C 5.6 */
/**********************************************/

/* Duplicate List of #defines for reference purposes:
 *
 * #define CON_PORT     2   Console / Keyboard Identification
 * #define KEY_PRESSED -1
 * #define NONE         0   May Be Omitted, Defined In AES.H
 * #define TRUE         1
 */
```

```
void pause_here()

{ int key_qty   = NONE ; /* Count Of Characters Pressed On Keyboard */
  int shiftcode = NONE ;        /* User Presses SHIFT, CTRL, or ALT */
  int mousecode = NONE ;                      /* Mouse Button Press */
  int mouse_x ;                       /* Mouse X Position On Screen */
  int mouse_y ;                       /* Mouse Y Position On Screen */

  long kb_status  = (long)NONE ;  /* Is Keypress Waiting For Input? */
  long user_input = (long)NONE ;  /* Information Received From User */

  char *key_message = "Press <any button> on the keyboard or
mouse.\r\n" ;

       /* Note: v_enter_cur is NOT USED since it clears the screen, */
           /* defeating the purpose of stop-and-look-at-the-output. */

  v_hide_c(  scrn_handle ) ;   /* Remove Mouse Pointer From Display */
  v_curtext( scrn_handle, key_message ) ;      /* Message On Screen */
  v_show_c(  scrn_handle, TRUE ) ;         /* Restore Mouse Pointer */

                                  /* Loop Waits For User To Respond */
  while( key_qty == NONE && shiftcode == NONE && mousecode == NONE )
  {                                           /* Check For Keypress */
     kb_status = Bconstat( CON_PORT ) ;     /* Is Keypress Waiting? */
     if( kb_status == (long)KEY_PRESSED )               /* If so... */
       { key_qty++ ;        /* Character Waiting Increments Counter */
         user_input = Bconin( CON_PORT ) ;        /* Clear Keypress */
       }
                    /* Check For Shift, Control, Alternate Keypress */

     vq_key_s( scrn_handle, &shiftcode ) ;

                                    /* Check For Mouse Button Press */
     vq_mouse( scrn_handle, &mousecode, &mouse_x, &mouse_y ) ;
  }
}
```

The Relationship Stress Test (For Guys) Measures Her Commitment
http://www.tmisnet.com/~strads/stress/index.html
Some Believe The Theory Of Dichotomy, But The Rest Of Us Do Not.

```
From -1690006187763487729
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,e0adbd549726a1bb
X-Google-Attributes: gid100493,public
X-Google-Thread: 1115ce,e0adbd549726a1bb
X-Google-Attributes: gid1115ce,public
From: lvzon@bio.vu.nl (Levien van Zon)
Subject: Re: Press Any Key....
```

Date: 1998/03/15
Message-ID: <6egcng$r5g@balaena.bio.vu.nl>#1/1
X-Deja-AN: 334326576
Distribution: world
References: <35074ac0.410905@news.tmisnet.com>
Followup-To: comp.sys.atari.programmer,comp.sources.atari.st
Organization: VU Biology, Amsterdam, The Netherlands
Newsgroups: comp.sys.atari.programmer,comp.sources.atari.st

George Crissman (strads@tmisnet.com) wrote:

: There was a recently-posted request that
: programs should stop after printing text
: to the screen (when exiting) so the user
: could view the information before the
: desktop reappears.

GEM programs should _never_ output text directly to the screen, and .TOS programs should _not_ pause after doing text output. This is _very_ annoying for people who use a shell (these people use lots of .TOS and .TTP programs as well). If a program does want to pause, it should definately not circumvent the GEMDOS character input routines, coz' the program's input/output is often redirected into a pipe!
Pause should be handled by whichever program runs the .TOS program, however, the standard desktop isn't that smart...
Maybe a good idea would be to check the extension of the program (in argv[0]) and if it is .tos or .ttp, pause after execution, but if there's no extension (ie. it's probably on a Minix FS) just quit without pause.
Or something like that... :)

: A compiled GEM program containing this
: function will be posted as "Two Bit Tutorial
: Number 1" (probably in the binaries group)
: soon. It will NOT need a resource (*.RSC)
: file for proper operation, so don't worry
: that it doesn't have one! (You can rename
: it PAUSE.TOS, if the lack of a resource
: file causes problems.)

Why would a GEM program want to do text output to the screen anyway?

     -=Levjenno

Levien van Zon (lvzon@bio.vu.nl)
MALEB Homepage: http://huizen.dds.nl/~levjenno/
(Atari/Amiga/Linux/Therapy?/Red Dwarf/etc.)

Whatever is not nailed down is mine. What I can pry loose is not nailed down.

-- Collis P. Huntingdon

From -6892643754943078287
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,e0adbd549726a1bb
X-Google-Attributes: gid100493,public
X-Google-Thread: 1115ce,e0adbd549726a1bb
X-Google-Attributes: gid1115ce,public
From: strads@tmisnet.com (George Crissman)
Subject: Re: Press Any Key....
Date: 1998/03/17
Message-ID: <350f0534.934129@news.tmisnet.com>#1/1
X-Deja-AN: 335060866
References: <35074ac0.410905@news.tmisnet.com> <6egcng$r5g@balaena.bio.vu.nl>
Distribution: world
Organization: MaximumAccess Internet Services
Newsgroups: comp.sys.atari.programmer,comp.sources.atari.st

On 15 Mar 1998 11:06:24 GMT, lvzon@bio.vu.nl (Levien van Zon) wrote:
>George Crissman (strads@tmisnet.com) wrote:

>GEM programs should _never_ output text
>directly to the screen,

Very true, but a lot of programs do it anyway --
usually during an initialization failure. The
original request for a TSR program that would
stop the computer AFTER a program exit and BEFORE
the desktop was redisplayed was finally satisfied
with TOSX (Atari GEMINI CD).

> and .TOS programs should _not_ pause after doing
> text output.

If they print to the screen as they exit, the text
is cleared so quickly that it cannot be read. The
pause is important!

> This is _very_ annoying for people who use a shell
>(these people use lots of .TOS and .TTP programs
>as well). If a program does want to pause, it should
>definately not circumvent the GEMDOS character input
>routines, coz' the program's input/output is often
>redirected into a pipe!

Good point.

> -=Levjenno

-- George Crissman
-- strads@tmisnet.com

The Relationship Stress Test (For Guys) Measures Her Commitment
http://www.tmisnet.com/~strads/stress/index.html
Some Believe The Theory Of Dichotomy, But The Rest Of Us Do Not.

From -9048610454298862993
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 1115ce,e0adbd549726a1bb
X-Google-Attributes: gid1115ce,public
X-Google-Thread: 100493,e0adbd549726a1bb
X-Google-Attributes: gid100493,public
From: lvzon@bio.vu.nl (Levien van Zon)
Subject: Re: Press Any Key....
Date: 1998/03/19
Message-ID: <6eqnge$h83@balaena.bio.vu.nl>#1/1
X-Deja-AN: 335529183
References: <35074ac0.410905@news.tmisnet.com> <6egcng$r5g@balaena.bio.vu.nl>
<350f0534.934129@news.tmisnet.com>
Followup-To: comp.sys.atari.programmer,comp.sources.atari.st
Distribution: world
Organization: VU Biology, Amsterdam, The Netherlands
Newsgroups: comp.sys.atari.programmer,comp.sources.atari.st

George Crissman (strads@tmisnet.com) wrote:

: > and .TOS programs should _not_ pause after doing
: > text output.

: If they print to the screen as they exit, the text
: is cleared so quickly that it cannot be read. The
: pause is important!

Yep, but the caller application (eg. the desktop) should take care of that
if needed. Unfortunately, as I said before, the GEM Desktop is not very
bright... (replacement desktops usually do have this option)
It may be possible to install a "viewer" program for .TOS and .TTP files
in the DESKTOP.INF. This program could ask for a commandline if needed
(maybe even using XARG, so we can get a decent commandline length!),
run the "client" program with the specified commandline, and pause after the
"client" has quit.

Or use the utility you mentioned. ;-)

     -=Levien

Levien van Zon (lvzon@bio.vu.nl)

MALEB Homepage: http://huizen.dds.nl/~levjenno/
(Atari/Amiga/Linux/Therapy?/Red Dwarf/etc.)

"I have two very rare photographs: one is a picture of Houdini locking
his keys in his car; the other is a rare photograph of Norman Rockwell
beating up a child."
         -- Steven Wright

From 5779691605449416096
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,5ad97cc71f78cbd9,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 10c818,5ad97cc71f78cbd9,start
X-Google-Attributes: gid10c818,public
X-Google-Thread: 100493,5ad97cc71f78cbd9,start
X-Google-Attributes: gid100493,public
From: "Chucky" <Chucky@breathemail.net>
Subject: Atari ST Games for sale
Date: 1999/06/04
Message-ID: <3758527c@news2.vip.uk.com>#1/1
X-Deja-AN: 487118024
X-Report: Report abuse to abuse@vip.uk.com
X-Original-NNTP-Posting-Host: 194.176.194.174
X-Trace: 8 Jun 1999 17:08:16 GMT, 10.250.101.2
X-MSMail-Priority: Normal
X-Priority: 3
Newsgroups: alt.games.atari,comp.sources.atari.st,comp.sys.atari.st
X-MimeOLE: Produced By Microsoft MimeOLE V5.00.2014.211

All boxed as new with manuals etc.

F19 Stealth Fighter
Microprose Grand Prix
Knights of the Sky
M1 Tank Platoon
Microprose Golf
Monkey Island adventure
Indiana Jones Last Crusade adventure
Midwinter II
Epic
Stunt Car Racer
Armour Geddon
Powermonger

Would like to sell as a job lot but may consider splitting
No serious offer refused, E-mail me for details.

From 7076155726975525196
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,1cb821b90d17206d,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 114bf5,1cb821b90d17206d,start
X-Google-Attributes: gid114bf5,public
X-Google-Thread: 100493,1cb821b90d17206d,start
X-Google-Attributes: gid100493,public
From: "Martijn" <martijn.kruger@solvay.com>
Subject: *** Which EPROM for TOS 2.06 in ST ??? ****
Date: 1999/06/09
Message-ID: <01beb262$4a631860$ee8ffb96@we0w3622.DUPHAR.SOLVAY.COM>#1/1
X-Deja-AN: 487441938
Organization: AT&T Communications Services Nederland B.V.
Newsgroups: comp.sources.atari.st,comp.sys.atari.st,comp.sys.atari.st.misc

Hi there !

I have an ATIN upgrade card (TOS) for my atari 1040.
The only thing that's left to do now is burn the EPROMS

But.... which EPROMS do I need for burning ?
Is there anyone who can send me the program for splitting the ROM image?
(the odd- and even story!)

Please reply to martijn.kruger@solvay.com

Greetings,
Martijn.

From 7270465494389464398
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,1cb821b90d17206d
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,1cb821b90d17206d
X-Google-Attributes: gid100493,public
X-Google-Thread: 114bf5,1cb821b90d17206d
X-Google-Attributes: gid114bf5,public
From: "Martin Graiter" <graiter@starmail.com>
Subject: SV: *** Which EPROM for TOS 2.06 in ST ??? ****
Date: 1999/06/10
Message-ID: <7jmrornmj3@cubacola.tninet.se>#1/1

X-Deja-AN: 487688818
References: <01beb262$4a631860$ee8ffb96@we0w3622.DUPHAR.SOLVAY.COM>
X-MimeOLE: Produced By Microsoft MimeOLE V5.00.2014.211
X-Complaints-To: abuse@algo.net
X-Trace: cubacola.tninet.se 928969307 24275 195.100.253.180 (9 Jun 1999 23:01:47 GMT)
Organization: Telenordia
NNTP-Posting-Date: 9 Jun 1999 23:01:47 GMT
Newsgroups: comp.sources.atari.st,comp.sys.atari.st,comp.sys.atari.st.misc

Martijn <martijn.kruger@solvay.com> skrev i diskussionsgruppsmeddelandet:01beb262$4a631860$ee8ffb96@we0w3622.DUPHAR.SOLVAY.COM...

> Hi there !
>
> I have an ATIN upgrade card (TOS) for my atari 1040.
> The only thing that's left to do now is burn the EPROMS
>
> But.... which EPROMS do I need for burning ?
> Is there anyone who can send me the program for splitting the ROM image?
> (the odd- and even story!)

The software for Junior Prommer fixes this, and also Pinatubo. I wrote a PRG for this a long time ago, but can't seem to find it. Pinatubo is freeware I believe.

The EPROMs should probably be the 27C010 if your sockets are 32 pin. You can use almost any kind if you care to make extra circuitry. A 74LS139 can split the CE (Chip Enable) signals to drive four, eight or sixteen EPROMs.

MG

From -7993040343676517200
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,4dc377ef918cce84,start
X-Google-Attributes: gid100493,public
X-Google-Thread: 102481,4dc377ef918cce84,start
X-Google-Attributes: gid102481,public
X-Google-Thread: fd4c9,4dc377ef918cce84,start
X-Google-Attributes: gidfd4c9,public
X-Google-Thread: 1056fc,4dc377ef918cce84,start
X-Google-Attributes: gid1056fc,public
X-Google-Thread: f8760,4dc377ef918cce84,start
X-Google-Attributes: gidf8760,public

From: "fozman" <fozman@breathemail.net>
Subject: Classic video game ROM's
Date: 2000/02/20
Message-ID: <38b058c0_3@news2.vip.uk.com>#1/1
X-Deja-AN: 587661138
X-Report: Report abuse to abuse@vip.uk.com
X-Original-NNTP-Posting-Host: 194.176.194.174
X-Trace: 20 Feb 2000 21:12:33 GMT, 10.250.101.2
Newsgroups:
comp.sources.atari,comp.sources.atari.st,comp.sys.atari,comp.sys.atari.8bit,comp.s
ys.atari.advocacy
X-MimeOLE: Produced By Microsoft MimeOLE V4.72.3110.3

Check out Rom Dumper for all the classic Atari System ROM's at
http://www.romdumper.free-online.co.uk/

Roms available for :-

Atari 2600
Atari 5200
ColecoVision
Master System
MSX
TG16
Vectrex

Atari 7800 coming soon

From 1594536285497654971
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,a2cf19543f616143,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: fa0ea,a2cf19543f616143,start
X-Google-Attributes: gidfa0ea,public
X-Google-Thread: 111655,a2cf19543f616143,start
X-Google-Attributes: gid111655,public
X-Google-Thread: f79d4,a2cf19543f616143,start
X-Google-Attributes: gidf79d4,public
X-Google-Thread: 100493,a2cf19543f616143,start
X-Google-Attributes: gid100493,public
X-Google-Thread: fd976,a2cf19543f616143,start
X-Google-Attributes: gidfd976,public
X-Google-Thread: fbdf7,a2cf19543f616143,start
X-Google-Attributes: gidfbdf7,public
X-Google-ArrivalTime: 2001-05-17 22:56:06 PST
Path: archiver1.sj.google.com!newsfeed.google.com!sn-xit-02!supernews.com!
news.gv.tsc.tdk.com!falcon.america.net!sunqbc.risq.qc.ca!news.maxwell.syr.edu!
newsfeed.skycache.com!Cidera!portc01.blue.aol.com!news.stealth.net!news-
east.rr.com!cyclone-west.rr.com!news.rr.com!news-west.rr.com!

lsnws01.we.mediaone.net!cyclone-LA3.rr.com!typhoon.we.rr.com.POSTED!not-for-mail
Message-ID: <3B04B886.622B48C4@mediaone.net>
From: Eddiesax <eddiesax@mediaone.net>
X-Mailer: Mozilla 4.08 [en] (Win98; I)
MIME-Version: 1.0
Newsgroups:
maus.computer.atari.hardware,comp.sources.atari.st,comp.sys.atari.st,ctdl.sys.atar
i.st,z-netz.rechner.atari.hardware,z-netz.rechner.atari.software,zer.z-
netz.rechner.atari.hardware
Subject: FA Atari St Computers & Software on EBay
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
Lines: 20
Date: Fri, 18 May 2001 05:53:40 GMT
NNTP-Posting-Host: 66.74.12.67
X-Complaints-To: abuse@mediaone.net
X-Trace: typhoon.we.rr.com 990165220 66.74.12.67 (Thu, 17 May 2001 22:53:40 PDT)
NNTP-Posting-Date: Thu, 17 May 2001 22:53:40 PDT
Organization: Road Runner
Xref: archiver1.sj.google.com maus.computer.atari.hardware:2063
comp.sys.atari.st:11434 ctdl.sys.atari.st:7 z-netz.rechner.atari.hardware:34

These items are currently available on EBay

Atari 1040 STE Personal Computer No Reserve
http://cgi.ebay.com/aw-cgi/eBayISAPI.dll?MfcISAPICommand=ViewItem&item=1237874033

Atari 1040 STF Personal Computer No Reserve
http://cgi.ebay.com/aw-cgi/eBayISAPI.dll?MfcISAPICommand=ViewItem&item=1237874043

Atari Computer Modem 1040 Mega ST No Reserve
http://cgi.ebay.com/aw-cgi/eBayISAPI.dll?MfcISAPICommand=ViewItem&item=1237874049

Notator Software W/ Unitor Atari Computer NR
http://cgi.ebay.com/aw-cgi/eBayISAPI.dll?MfcISAPICommand=ViewItem&item=1430942487

Sound Designer Software Atari Computer NR
http://cgi.ebay.com/aw-cgi/eBayISAPI.dll?MfcISAPICommand=ViewItem&item=1430942494

Atari Computer Modem 1040 Mega ST No Reserve
http://cgi.ebay.com/aw-cgi/eBayISAPI.dll?MfcISAPICommand=ViewItem&item=1237874049

From -6878885243159097270
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,8dcb3ed1a0dab1f7,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-06-03 12:10:12 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
agate!saturn!koreth@ssyx.ucsc.edu
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st

Subject: v02i041: fplib20 -- New floating-point library for Sozobon C part01/02
Keywords: shar, C
Message-ID: <7809@saturn.ucsc.edu>
Date: 3 Jun 89 19:10:12 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 1387
Approved: koreth@ssyx.ucsc.edu

Submitted-by: dbrooks@osf.org (David Brooks)
Posting-number: Volume 2, Issue 41
Archive-name: fplib20/part01

This is revision 2.0 of FPLIB, a full floating-point library for use
with Sozobon C on the Atari ST. It fixes some minor bugs in revision
1.0, improves the accuracy and speed of all floating point operations,
and includes better documentation.

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
# This is part 1 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
#
#    Run the following text with /bin/sh to create:
#      ATOF.C
#      DNOTES
#      EXP.C
#      FABS.C
#      FMOD.C
#      FPADD.S
#      FPCMP.S
#      FPDIV.S
#      FPLIB.H
#      FPMUL.S
#      FP_PRT.C
#      ITRIG.C
#      LIBM.uue
#      LOG.C
#      MATH.H
#      POW.C
#      README
#      RNOTES
#      SPECS
#      SQRT.C
#      TRIG.C
#
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
sed 's/^X//' << 'SHAR_EOF' > ATOF.C &&
X/* ATOF function for Sozobon C.                        */
X/* Copyright = David Brooks, 1989 All Rights Reserved
```

```
         */
X
X/* Hack warning.  Hack warning.  Hack warning...
     */
X/* Somehow I feel scanf should call atof, not the other way around.
     */
X/* Guys???                                      */
X
Xfloat atof(s)
Xchar *s;
X{   float    res;
X
X    sscanf(s, "%g", &res);
X    return res;
X}
X
X_scandum()
X{
X    _scanf();          /* Force floating version to load */
X}
SHAR_EOF
chmod 0600 ATOF.C || echo "restore of ATOF.C fails"
sed 's/^X//' << 'SHAR_EOF' > DNOTES &&
X        Design notes for FPLIB Release 2.0
X        ==================================
X
X
XNUMBER REPRESENTATION
X---------------------
X
XThe floating point format used is Motorola's "Fast Floating Point":
X
X        ----------------------------------------
X        |    24 bits mantissa    |S| 7 bits exp |
X        ----------------------------------------
X
XThe mantissa has an implicit binary point just off to the left.  The
Xexponent is binary, stored excess-64.  In other words, the actual
Xmathematical exponent has 64 added, and the resulting 7-bit number is
Xunsigned.
X
XNegative numbers are stored as sign-and-magnitude.
X
XAll floating-point operations expect normalized numbers, and generate
Xnormalized numbers if the inputs are normalized.  A number is
Xnormalized if the most significant bit is 1 (except for 0.0, which is
Xstored as 32 bits of zero, never negative).  Also, a nonzero mantissa
Xwith a biased exponent of 0 is not normalized, letting us test the low
Xbyte or whole long as a test for 0.0.  This all means that the most
Xsignificant bit is actually redundant, and could be left off, giving
Xone extra bit of precision.  Oh well.
X
XSome of these routines make use of the assumption that all arguments
```

Xare normalized, and may generate slightly bogus results if that isn't
Xtrue. Release 1.0 assumed that a biased exponent of zero could have
Xa nonzero mantissa; that has been fixed.
X
XSome examples: 1.0 is hex 80000041 , or 0.5 * 2 ** 1 (binary 0.1,
Xexponent 1 + bias of 64). -0.75 is hex C00000C0 (note the sign bit is
Xinverted, but no other bit is).
X
X
XACCURACY AND LIMITS
X------------------
X
XThe accuracy is in the 24-bit mantissa, which gives between 7.22 and
X7.52 decimal digits of relative error. Decimal constants are given to
X8 digits, and the compiler is trusted to convert them right (it
Xgenerally does a reasonable job).
X
XThe range is in the exponent. The smallest nonzero and largest
Xpositive normalized numbers are ~5.4e-20 and ~9.2e18.
X
XRounding of ambiguous numbers uses round-to-even (see D. E. Knuth's
XSeminumerical Algorithms for a discussion on this). This means that
Xif the trailing portion is exactly one half of the least significant
Xbit, the lsb is forced to be the nearer even number. We often keep
Xseveral insignificant bits to be sure this is done correctly.
X
X
XCODING TRICKS
X-------------
X
XIt's quite normal in a library of this type to resort to bit-twiddling
Xin the internal representation. To help this, FPLIB.H defines a C
Xunion, thus:
X
Xunion   {
X   unsigned long ul;
X   float f;
X   char sc[4];
X   };
X
XThe number can be looked at as a floating or integer long (for handling
Xbits in the mantissa), the sign is the sign of sc[3], and the exponent
Xis the rest of sc[3]. Other shortcuts are commented in the inividual
Xroutines.
X
X
XMATHEMATICAL INFORMATION
X------------------------
X
XWell, this is mostly what I remember from my undergraduate days. Many
Xof these functions have a precise mathematical formula or algorithm
Xthat will give the result, but is too slow for practical use. Instead,
Xwe look for an approximation that is quick to compute, and gives us

```
Xjust enough precision and no more.  After all, why compute to 100
Xdigits accuracy when you can only represent less than 8?  Some of these
Xapproximations look very strange, but they all have sound mathematical
Xbases.  The name Chebyshev springs to mind.
X
XDedicated mathematicians have struggled for decades to bring you these
Xformulae and, just as important, a predictable accuracy of better than
X7.5 decimal digits.  Because of this, we can execute a known number of
Xoperations instead of loop-and-test (as you would in, for example, a
Xsimple Newton's iteration for square root).
X
XThe approximations are generally polynomials, usually avoid division, and
Xare valid only over a small range of the input parameter.  We use simple
Xmathematical identities to extend the range.  For example, the cyclic
Xnature of the trigonometric functions means we only need calculate in the
Xrange 0..pi/2.  For another example, we can do much of the square root
Xcalculation by the simple expedient of dividing the binary exponent by 2.
X
X
XRANDOM LIBRARY INFORMATION
X--------------------------
X
XRelease 1.0 of Sozobon C had a few problems in the floating support:
X
X- Two negative floating numbers compared wrong (returned > when it
X  should be <).  I corrected that.  However, the other routines don't
X  assume the correction has been installed in the library (which it
X  wasn't, at first...).
X
X- The library was not optimally ordered, and had to be scanned twice.
X  I straightened that out.
X
X- Add and subtract were truncating.  Rounding is generally a better idea.
X
Xatof() calls sscanf().  Ick.
X
X-------------------------------------------------------------------------------
X
XThese routines may be used without restriction, but I retain the copyright
Xon them, and the Only True Upgrades will be those I make.  If you have any
Ximprovements, please send them to me.
X
XMarch 1989
X
XDavid W. Brooks
X36 Elm Street
XMedfield, MA 02052
X
XBROOKS@CSSS-A.PRIME.COM
X{uunet,mit-eddie}!csss-a.prime.com!brooks
SHAR_EOF
chmod 0600 DNOTES || echo "restore of DNOTES fails"
sed 's/^X//' << 'SHAR_EOF' > EXP.C &&
```

```
X/* EXP and hyperbolic trig functions for Sozobon C.
         */
X/* Copyright = David Brooks, 1989 All Rights Reserved
         */
X
X#include <fplib.h>
X#include <errno.h>
X
X/* EXP:                                          */
X/*                                            */
X/* Uses the synthesis (2**n) * (2**[m/8]) * b**g
     */
X/* where b = 2**(1/8) and 8*n+m+g = arg/ln(b).  b**g has a cubic
     */
X/* approximation: source and accuracy unknown.
         */
X/*                                            */
X/* Beware a bug in the standard release: can't compare two negative
     */
X/* floating numbers.                                */
X
Xfloat exp(a) register float a;
X{   fstruct       res;
X    register int  aint;
X
X    static fstruct huge = {HUGE_AS_INT};
X    static float powtab[] = {1.0, 1.09050773, 1.1892071, 1.2968396,
X             1.41421356, 1.5422108, 1.68179283, 1.8340081};
X
X    if (a > 43.5)
X    {    errno = ERANGE;
X         return huge.f;
X    }
X
X    if ((a + 43.5) < 0.0)
X         return 0.0;                 /* Underflow */
X
X    if ((aint = (int)(a *= 11.5415603)) < 0)  /* 8/ln(2) */
X         --aint;                /* Correct mathematically */
X    a = (float)aint - a;             /* -(frac part)
     */
X
X    res.f = 1.0 + a * (-0.0866439378 + a * (0.003750577 +
X                  a * -0.11321783e-3));
X    res.sc[3] += aint >> 3;          /* Mult by 2**n
     */
X    return res.f * powtab[aint&7];       /* Mult by 2**[m/8]
     */
X}
X
X/* HYPERBOLIC FUNCTIONS: virtually free
     */
X
```

```
Xfloat sinh(a) float a;
X{   register float     ea = exp(a);
X
X    return 0.5 * (ea - (1.0 / ea));
X}
X
Xfloat cosh(a) float a;
X{   register float     ea = exp(a);
X
X    return 0.5 * (ea + (1.0 / ea));
X}
X
Xfloat tanh(a) float a;
X{   register float     e2a;
X
X    e2a = exp(a);
X    e2a = e2a * e2a;            /* exp-squared-a */
X    return (e2a - 1.0) / (e2a + 1.0);
X}
X
SHAR_EOF
chmod 0600 EXP.C || echo "restore of EXP.C fails"
sed 's/^X//' << 'SHAR_EOF' > FABS.C &&
X/* FABS, LDEXP, and FREXP functions for Sozobon C.
     */
X/* Copyright = David Brooks, 1989 All Rights Reserved
          */
X
X/* Mustn't allow the standard declaration of fabs()... */
X#define MATH_H
X
X#include <fplib.h>
X#include <errno.h>
X
X/* FABS: directly manipulate the sign bit, punning the type.  Actually
     */
X/* the argument and return are both float.
     */
X
Xunsigned long fabs(a)
Xunsigned long a;
X{   return a & 0xFFFFFF7FL;}
X
X/* LDEXP: again direct manipulation.                    */
X
Xfloat ldexp(a, n)
Xfstruct a;
Xint n;
X{
X/* "long" to handle int overflow/underflow... */
X    register long exp = (long)(a.sc[3] & EXP_MASK) + n;
X
X    if (exp <= 0L)
```

```
X        return 0L;             /* Underflow */
X
X    if (exp > 0x7FL)
X    {    errno = ERANGE;            /* Overflow */
X         a.ul |= 0xFFFFFF7FL;       /* Preserve sign */
X         return a.f;
X    }
X
X    a.sc[3] = (a.sc[3] & 0x80) | exp;
X    return a.f;
X}
X
X/* FREXP: split into convenient mantissa and exponent
         */
X
Xfloat frexp(a, ip)
Xfstruct a;
Xint *ip;
X{   if (a.sc[3] == 0)
X         *ip = 0;
X    else
X    {    *ip = (a.sc[3] & 0x7f) - BIAS;
X         a.sc[3] = (a.sc[3] & 0x80) | BIAS;    /* preserve sign,
X                                    force exponent */
X    }
X
X    return a.f;
X}
SHAR_EOF
chmod 0600 FABS.C || echo "restore of FABS.C fails"
sed 's/^X//' << 'SHAR_EOF' > FMOD.C &&
X/* CEIL, FLOOR, MODF, and FMOD functions for Sozobon C.
              */
X/* Copyright = David Brooks, 1989 All Rights Reserved
         */
X
X#include <fplib.h>
X
X/* This macro returns true if the number is so large that is has no
     */
X/* representable fraction.                         */
X
X#define ISLGINT(a)    (((a).sc[3] & EXP_MASK) > (BIAS + MANT_BITS))
X
X/* CEIL: smallest integral value not less than a.
     */
X
Xfloat ceil(a)
Xfstruct a;
X{   register float    aint;
X
X    if (ISLGINT(a))
X         return a.f;
```

```
X
X    if ((aint = (float)((long)a.f)) != a.f && a.sc[3] >= 0)
X         ++aint;
X
X    return aint;
X}
X
X/* FLOOR: largest integral value not greater than a.
         */
X
Xfloat floor(a)
Xfstruct a;
X{   register float     aint;
X
X    if (ISLGINT(a))
X         return a.f;
X
X    if ((aint = (float)((long)a.f)) != a.f && a.sc[3] < 0)
X         --aint;
X
X    return aint;
X}
X
X/* Just for once, we luck out: negative numbers behave as needed... */
X
Xfloat modf(a, ip)
Xfstruct a;
Xfloat *ip;
X{   long ipart;
X
X    *ip = (ISLGINT(a))?a.f:(float)(long)a.f;
X
X    return a.f - *ip;
X}
X
Xfloat fmod(n, d)
Xfloat n, d;
X{   fstruct quot;
X    long iquot;
X
X    quot.f = n / d;
X    if (ISLGINT(quot))
X         return 0.0;
X
X    iquot = (long)quot.f;
X    return n - (float)iquot * d;
X}
SHAR_EOF
chmod 0600 FMOD.C || echo "restore of FMOD.C fails"
sed 's/^X//' << 'SHAR_EOF' > FPADD.S &&
X* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X*
X* Permission is granted to anyone to use this software for any purpose
```

```
X* on any computer system, and to redistribute it freely, with the
X* following restrictions:
X* 1) No charge may be made other than reasonable charges for reproduction.
X* 2) Modified versions must be clearly marked as such.
X* 3) The authors are not responsible for any harmful consequences
X*    of using this software, even if they result from defects in it.
X*
X*
X* MODIFICATIONS:
X*
X* D.W.Brooks Mar 89   Implemented rounding, inlined one-call subroutines,
X*            and several other speedups
X*
X* Rounding in the ambiguous case (tail byte = 0x80) uses round-to-even, as
X* Knuth insists (TAOCP3SNA).
X*
X*  fpsub
X*
X    .globl   fpsub
X    .globl   _fpsub
Xfpsub:
X_fpsub:
X    eor.b    #$80,11(sp)   * negate B, fall through (can produce -0.0)
X*
X*  fpadd
X*
X    .globl   fpadd
X    .globl   _fpadd
Xfpadd:
X_fpadd:
X    move.l   d3,a1         * save d3
X    move.l   d4,a2         * and d4
X    move.l   4(sp),d0 * First arg
X    move.l   8(sp),d1 * Second arg
X    move.l   d0,d4         * Keeps the result's sign
X    moveq.l  #$7f,d3
X
X    move.b   d0,d2         * Compare signs only
X    eor.b    d1,d2
X    bpl  same_sign
X* different signs
X    and.b    d3,d0         * Force positive
X    and.b    d3,d1
X
X    move.l   d0,d2         * note d2 is copy of d0
X    cmp.b    d0,d1         * Compare magnitudes
X    bne  sk1
X    cmp.l    d0,d1
Xsk1:
X    ble  norev1
X    move.l   d1,d0         * swap d0/d1...d0 was saved in d2...
X    move.l   d2,d1
X    move.b   d0,d2         * ...now d2.b is copy of new d0.b
```

```
X    not.b    d4       * Invert sign
Xnorev1:
X* Begin unsigned subtract. Here we have (d0-d1), d0 >= d1, both positive.
X* d2.b has result exp.  Subtract 0.0 works without needing a special case.
X
X    move.b   d0,d3
X    sub.b    d1,d3         * diff of exps (in low byte)
X
X    clr.b    d0       * Extract mantissas
X    clr.b    d1
X
X    cmp.b    #25,d3
X    bgt  out      * Nonsignificant addend: d0 is correct mantissa
X    lsr.l    d3,d1         * Shift uses d3 mod 64
X    sub.l    d1,d0
X
X    beq  out0     * Zero: all done!
X    bmi  outsub        * Already normalized
X* normalize loop
Xnloop:
X    subq.b   #1,d2
X    ble  underfl
X    add.l    d0,d0
X    bpl  nloop
X
Xoutsub:
X    add.l    #$80,d0       * Force a rounding step.  This can't overflow.
X                 * (rum-ti-rum...can it?)
X*
X* Common exit point for checking rounding after adding 0x80.  If the tail
X* was not 0x80, it's already the correct result.  Otherwise, the tail is now
X* 0, and we implement the round-to-even.
X*
Xoutround:
X    tst.b    d0
X    bne  out
X    and.w    #$FE00,d0
X*
X* Common exit point.  d0 has the mantissa, d2 the exponent and d4 the sign.
X*
Xout:
X    move.b   d2,d0
X    and.b    #$80,d4
X    or.b d4,d0         * fix sign
X*
X* Come here when d0 has the result
X*
Xout0:
X    move.l   a1,d3         * restore
X    move.l   a2,d4
X    rts
X
X* underflow
```

```
Xunderfl:
X    moveq.l  #0,d0
X    bra  out0
X*
X* same signs: do add.
X*
Xsame_sign:
X    and.b    d3,d0         * Force both positive
X    and.b    d3,d1
X
X    move.l   d0,d2
X    cmp.b    d0,d1         * Compare magnitudes
X    bne  sk2
X    cmp.l    d0,d1
Xsk2:
X    ble  norev2
X    move.l   d1,d0         * Swap d0/d1...
X    move.l   d2,d1
X    move.b   d0,d2         * ...and d2.b is a copy of d0.b
Xnorev2:
X* Begin unsigned add: (d0+d1), d0 >= d1
X    sub.b    d1,d0
X    move.b   d0,d3         * diff of exps
X
X    clr.b    d0       * Extract mantissas
X    clr.b    d1
X
X    cmp.b    #25,d3
X    bge  out      * Nonsignificant addend: d0 has result mantissa
X    lsr.l    d3,d1
X    add.l    d1,d0
X    bcc  around
X    roxr.l   #1,d0         * Mantissa overflowed the word
X    addq.b   #1,d2
X    bmi  oflo     * Exponent too hot to handle
X
Xaround:               * Round the fraction.
X    add.l    #$80,d0       * This rounds unless d0.b was 0x80
X    bcc  outround * No overflow, but do final rounding
X    roxr.l   #1,d0         * Actually loads 0x800000nn
X    addq.b   #1,d2         * Compensate
X    bpl  out      * Ok if no exponent overflow
X
Xoflo:
X    moveq.l  #$ffffffff,d0 * Handle exponent overflow, maintaining sign
X    moveq.l  #$7f,d2
X    bra  out
X
X    .end
SHAR_EOF
chmod 0600 FPADD.S || echo "restore of FPADD.S fails"
sed 's/^X//' << 'SHAR_EOF' > FPCMP.S &&
X* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
```

```
X*
X* Permission is granted to anyone to use this software for any purpose
X* on any computer system, and to redistribute it freely, with the
X* following restrictions:
X* 1) No charge may be made other than reasonable charges for reproduction.
X* 2) Modified versions must be clearly marked as such.
X* 3) The authors are not responsible for any harmful consequences
X*    of using this software, even if they result from defects in it.
X*
X* MODIFICATIONS:
X*
X* D.W.Brooks 3/5/89   Fixed fpcmp (when both -ve) and fpneg (when zero)
X*
X
X    .globl   fpcmp
X    .globl   _fpcmp
Xfpcmp:
X_fpcmp:
X    move.l   4(sp),d0
X    move.l   8(sp),d1
X* Start DWB
X    move.b   d0,d2         * See if both signs are negative
X    and.b    d1,d2
X    bpl  L0
X    move.l   d0,d2         * They are: switch arguments so the rest works
X    move.l   d1,d0
X    move.l   d2,d1
XL0:
X* End DWB
X    cmp.b    d1,d0
X    bne  L1
X    cmp.l    d1,d0
XL1:
X    rts
X
X    .globl   fpneg
X    .globl   _fpneg
Xfpneg:
X_fpneg:
X    move.l   4(sp),d0
X
X* DWB removed: tst.b  d0 (unnecessary to test for 0.0)
X
X    beq  L2
X    eor.b    #$80,d0
XL2:
X    rts
SHAR_EOF
chmod 0600 FPCMP.S || echo "restore of FPCMP.S fails"
sed 's/^X//' << 'SHAR_EOF' > FPDIV.S &&
X* FPDIV.S -- floating point divide
X* Copyright = David Brooks, 1989 All Rights Reserved
X*
```

```
X* This version works by a normal bitwise shift/subtract loop, with separate
X* handling of the exponent and a final normalization.  26 bits are computed,
X* to allow one normalization and one rounding step.  A certain amount of
X* black magic and handwaving are involved, but please note that at no time
X* do any bits leave the confines of the computer.
X*
X* Round-to-even is used for the ambiguous rounding case.
X*
X* float fpdiv(num, denom); (_fpdiv for explicit call; fpdiv for compiled /)
X
X    .globl   fpdiv
X    .globl   _fpdiv
X
Xfpdiv:
X_fpdiv:
X    move.l   d3,a1             * We don't need no wimpy stack frame
X    moveq.l  #0,d0
X    move.l   4(sp),d1
X    beq  exit          * Numerator 0 - all done
X    move.l   8(sp),d2
X    beq  div0          * Divide by 0 - overflow
X    clr.b    d1            * Work with mantissas
X    clr.b    d2
X    move.l   #$2000000,d3      * Position of msb of 26-bit field
Xcmpbit:
X    cmp.l    d2,d1             * Compare against new divisor
X    blo  nobit
X    add.l    d3,d0             * Add in this bit
X    sub.l    d2,d1             * and adjust
Xnobit:
X    lsr.l    #1,d2
X    lsr.l    #1,d3
X    bne  cmpbit            * Done 26 bits?
X
X    lsl.l    #6,d0             * Reposition
X    tst.l    d1            * If there was a remainder...
X    beq  doexp
X    addq.l   #2,d0             * record a memory of it.  The 2 bit
X                        * .won't be lost by either of the
X                        * ..subsequent normalizations
Xdoexp:
X    moveq.l  #$7F,d1           * Calculate new exponent
X    move.l   d1,d2
X    and.b    7(sp),d1
X    and.b    11(sp),d2
X    sub.w    d2,d1
X    add.w    #$41,d1           * Adjust
X    tst.l    d0            * Check for already normalized or zero
X    bmi  normok
Xnormloop:
X    subq.w   #1,d1             * Do one normalize step (there can't
X    add.l    d0,d0             * .be more)
Xnormok:
```

```
X    add.l    #$80,d0           * Round up in most cases
X    bcc  round1
X    roxr.l   #1,d0             * The rounding caused overflow, sigh
X    addq.w   #1,d1
Xround1:
X    tst.b    d0            * See if trailer was exactly 0x80
X    bne  ckexp
X    and.w    #$FE00,d0     * It was: round to even
X
Xckexp:                     * Check exponent for sanity
X    tst.w    d1
X    ble  underflow
X    cmp.w    #$7F,d1
X    bgt  overflow
Xsetexp:
X    move.b   d1,d0             * Set exponent
X
X    move.b   7(sp),d1      * Get signs
X    move.b   11(sp),d2
X    eor.b    d2,d1             * Form new sign
X    and.b    #$80,d1           * Extract it
X    or.b d1,d0
Xexit:
X    move.l   a1,d3
X    rts
Xdiv0:
Xoverflow:
X    moveq.l  #$FFFFFFFF,d0
X    moveq.l  #$7F,d1
X    bra  setexp
X
Xunderflow:
X    moveq.l  #0,d0
X    bra  exit
X
X    .end
SHAR_EOF
chmod 0600 FPDIV.S || echo "restore of FPDIV.S fails"
sed 's/^X//' << 'SHAR_EOF' > FPLIB.H &&
X/* Common definitions for FPLIB                        */
X/* Copyright = David Brooks, 1989 All Rights Reserved
         */
X
X#ifndef  MATH_H
X#include <math.h>     /* Now they're all defined */
X#endif
X
X/* Some facts about the representation
     */
X
X#define BIAS      0x40
X#define MANT_BITS 24
X#define EXP_MASK  0x7F
```

```
X#define HUGE_AS_INT   0xFFFFFF7F
X
X/* Redefine a 32-bit float number.  Can initialize with unsigned long.
     */
X
Xtypedef union {
X    unsigned long ul;
X    float f;
X    char sc[4];            /* Assuming char signed */
X    } fstruct;
SHAR_EOF
chmod 0600 FPLIB.H || echo "restore of FPLIB.H fails"
sed 's/^X//' << 'SHAR_EOF' > FPMUL.S &&
X* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X*
X* Permission is granted to anyone to use this software for any purpose
X* on any computer system, and to redistribute it freely, with the
X* following restrictions:
X* 1) No charge may be made other than reasonable charges for reproduction.
X* 2) Modified versions must be clearly marked as such.
X* 3) The authors are not responsible for any harmful consequences
X*    of using this software, even if they result from defects in it.
X*
X* MODIFICATION:
X*
X* D.W.Brooks Apr 89   Tightened up some coding, and looked after some
X*            trailing bits for the sake of textbook rounding.
X*
X*   fpmul
X*
X    .globl   _fpmult
X    .globl   _fpmul
X    .globl   fpmul
Xfpmul:
X_fpmul:
X_fpmult:
X    move.l   d3,a0         * save scratch registers
X    move.l   d4,a1
X    move.l   d5,a2
X    move.l   4(sp),d0
X    move.l   8(sp),d1
X
X    move.l   #$7F,d2       * Calculate the presumed exponent
X    move.l   d2,d3
X    and.w    d0,d2
X    beq  ret      * Mult by 0: d0 is correct
X    and.w    d1,d3
X    beq  ret0     * Mult by 0: set d0
X    add.w    d3,d2
X    sub.w    #$40,d2       * Remove the extra bias
X
X    clr.b    d0       * Calculate the absolute mantissa product
X    clr.b    d1
```

```
X    move.w   d0,d3         * Get least significant bits of result
X    mulu.w   d1,d3
X    swap d3       * d3 range now 0...fe01
X
X    move.w   d0,d4
X    move.w   d1,d5
X    swap d0
X    swap d1
X    mulu d0,d5         * calculate inner portion
X    mulu d1,d4
X    add.l    d3,d4         * no carry since d3 <= fe01 && d4 <= fffe0001
X    add.l    d4,d5
X    bcc  nocar1
X    move.w   d5,d3         * Save trailing bits
X    move.w   #1,d5
X    bra  t1
Xnocar1:
X    move.w   d5,d3
X    clr.w    d5
Xt1:
X    swap d5
X
X    mulu d1,d0         * calculate most significant part
X    add.l    d5,d0
X    bcc  nocar2
X    roxr.l   #1,d0
X    addq.w   #1,d2         * Normalize down
Xnocar2:
X    bmi  norm
X    add.l    d0,d0         * only need at most 1 shift: started norm AB
X    subq.w   #1,d2
Xnorm:
X    add.l    #$80,d0       * Round uppppp....
X    bcc  nocar3
X    roxr.l   #1,d0
X    addq.w   #1,d2
Xnocar3:
X    tst.b    d0       * Check if trailer was exactly 0x80 (now 0x00)
X    bne  rebuild
X    tst.w    d3       * Check back with spare bits
X    bne  rebuild
X    and.w    #$FE00,d0* Round to even
Xrebuild:
X    tst.w    d2       * Reconstruct the number.  First test overflow
X    ble  underflow* Could be mult by 0
X    cmp.w    #$7F,d2
X    bgt  overflow
Xexpsign:
X    move.b   d2,d0         * Stuff exponent in
X    move.b   7(sp),d1 * Calculate sign
X    move.b   11(sp),d2
X    eor.b    d1,d2
X    and.b    #$80,d2
```

```
X    or.b d2,d0
Xret:
X    move.l   a2,d5
X    move.l   a1,d4
X    move.l   a0,d3
X    rts
X
Xret0:
Xunderflow:
X    move.l   #0,d0         * Underflow, return 0
X    bra  ret
X
Xoverflow:
X    move.l   #$7F,d2       * Overflow, return +/- huge
X    move.l   #$FFFFFFFF,d0
X    bra  expsign
X
X    .end
SHAR_EOF
chmod 0600 FPMUL.S || echo "restore of FPMUL.S fails"
sed 's/^X//' << 'SHAR_EOF' > FP_PRT.C &&
X/* Modified:                              */
X/* DWB Apr 89     Handle unnormalized numbers          */
X/*       Fix exponent if .9... overflows           */
X
X#define MAXEC     40
X
Xstatic char *f_buf;
Xstatic int f_upper;
X
Xstatic char ec_buf[MAXEC+10];
Xstatic int ec_sign, ec_exp;
X
Xfp_print(x, fmtc, prec, ptmp)
Xfloat x;
Xchar *ptmp;
X{   register long lx, m;
X    register int  e;
X
X/* These routines have trouble with unnormalized numbers. */
X
X    lx = *(long*)&x;
X    if ((m = lx & 0xffffff00L) == 0 || (e = lx & 0x7f) == 0)
X         x = 0.0;
X    else if (m > 0L)
X    {    do
X         {    if (--e == 0)
X              {    x = 0.0;
X                   goto uflo;
X              }
X              m <<= 1;
X         } while (m > 0L);
X         *(long*)&x = m | e | (lx & 0x80L);
```

```
Xuflo:
X    }
X
X    f_buf = ptmp;
X    f_upper = 0;
X
X    switch (fmtc) {
X    case 'E':
X         f_upper = 1;
X    case 'e':
X         e_print(x, prec);
X         break;
X    case 'F':
X    case 'f':
X         f_print(x, prec);
X         break;
X    case 'G':
X         f_upper = 1;
X    case 'g':
X         g_print(x, prec);
X         break;
X    }
X}
X
Xstatic
Xe_print(x, prec)
Xfloat x;
X{
X    int nsig;
X    register char *p;
X
X    if (prec < 0)
X         nsig = 7;
X    else
X         nsig = prec+1;
X
X    ec_pr(x, nsig, 0);
X
X    p = f_buf;
X    if (ec_sign)
X         *p++ = '-';
X    *p++ = ec_buf[0];
X    *p++ = '.';
X    if (nsig > 1)
X         strcpy(p, &ec_buf[1]);
X    p += strlen(p);
X    *p++ = f_upper ? 'E' : 'e';
X    ec_exp--;
X    if (ec_exp < 0) {
X         *p++ = '-';
X         ec_exp = -ec_exp;
X    }
X    if (ec_exp < 10)
```

```
X         *p++ = '0';
X    sprintf(p, "%d", ec_exp);
X}
X
Xstatic
Xf_print(x, prec)
Xfloat x;
X{
X    int nsig, nz, i;
X    register char *p;
X
X    if (prec < 0)
X         nsig = 6;
X    else
X         nsig = prec;
X
X    ec_pr(x, -nsig, 0);
X
X    p = f_buf;
X    if (ec_sign)
X         *p++ = '-';
X    if (ec_exp < 1) {
X         *p++ = '0';
X    } else {
X         strncpy(p, ec_buf, ec_exp);
X         p += ec_exp;
X    }
X    if (prec != 0 || nsig)
X         *p++ = '.';
X    if (nsig == 0) {
X         *p = 0;
X         return;
X    }
X
X    if (ec_exp < 0) {
X         nz = -ec_exp;
X         if (nz > nsig)
X              nz = nsig;
X         for (i=0; i<nz; i++)
X              *p++ = '0';
X         nsig -= nz;
X         if (nsig > 0) {
X              strncpy(p, ec_buf, nsig);
X              p += nsig;
X         }
X         *p = 0;
X    } else {
X         strcpy(p, &ec_buf[ec_exp]);
X    }
X}
X
Xstatic
Xg_print(x, nsig)
```

```
Xfloat x;
X{
X    int prec;
X
X    if (nsig < 0)
X         nsig = 6;
X    if (nsig < 1)
X         nsig = 1;
X
X    ec_pr(x, 1, 1);
X
X    if (ec_exp < -3 || ec_exp > nsig)
X         e_print(x, nsig-1);
X    else {
X         prec = nsig - ec_exp;
X         f_print(x, prec);
X    }
X}
X
X/*
X * given x, ndig
X *  if ndig is > 0, indicates number of significant digits
X *  else -ndig is number of digits we want to the right of dec. pt.
X * return the following:
X *  appropriate number of digits of significance in ec_buf
X *  ec_sign true if x was negative
X *  ec_exp indicates the decimal point relative to leftmost digit
X */
Xstatic
Xec_pr(x, ndig, trunc)
Xfloat x;
X{
X    int isneg;
X    int nhave;
X    long part;
X    int exp, newexp;
X    int ndigcpy = ndig;
X    float rem;
X    char tbuf[20];
X
X    /* ndig must be >= 1 and <= MAXEC */
X    if (x < 0.0) {
X         isneg = 1;
X         x = -x;
X    } else
X         isneg = 0;
X
X    /* get some digits */
X    somedig(x, &part, &rem, &exp);
X
X    sprintf(ec_buf, "%ld", part);
X    nhave = strlen(ec_buf);
X    exp = nhave + exp;
```

```
X
X    if (ndig <= 0) {
X         ndig = -ndig;
X         ndig += exp;
X    }
X
X    if (ndig < 1)
X         ndig = 1;
X    else if (ndig > MAXEC)
X         ndig = MAXEC;
X
X    /* get some more digits */
X    while (nhave < ndig+1 && nhave < MAXEC) {
X         if (rem == 0.0) {
X              while (nhave < ndig+1)
X                   ec_buf[nhave++] = '0';
X              ec_buf[nhave] = 0;
X              break;
X         }
X
X         x = rem;
X         somedig(x, &part, &rem, &newexp);
X
X         sprintf(tbuf, "%ld", part);
X         newexp = strlen(tbuf) + newexp;
X         while (newexp++)
X              ec_buf[nhave++] = '0';
X         strcpy(&ec_buf[nhave], tbuf);
X         nhave = strlen(ec_buf);
X    }
X
X    if (fround(ndig, trunc))
X    {    if (ndigcpy <= 0)
X              strcpy(ec_buf+ndig, "0");
X         exp++;
X    }
X
X    ec_sign = isneg;
X    ec_exp = exp;
X}
X
Xstatic
Xfround(n, trunc)
X{
X    char *p;
X    int oflo = 0;
X
X    p = &ec_buf[n];
X    if (*p >= '5' && !trunc) {
X         p--;
X         while (p >= ec_buf) {
X              *p += 1;
X              if (*p < '9')
```

```
X                  goto done;
X             *p = '0';
X             p--;
X        }
X        ec_buf[0] = '1';
X        oflo = 1;
X    }
Xdone:
X    ec_buf[n] = 0;
X    return oflo;
X}
X
Xstatic
Xsomedig(x, lp, remp, expp)
Xfloat x;
Xlong *lp;
Xfloat *remp;
Xint *expp;
X{
X    int bexp, dexp;
X    long ipart;
X    float rem;
X
X    bexp = fgetexp(x);
X    dexp = 0;
X
X    while (bexp > 31) {
X        x *= 1E-3;
X        dexp += 3;
X        bexp = fgetexp(x);
X    }
X    while (bexp < 10) {
X        x *= 1E3;
X        dexp -= 3;
X        if (dexp < -24) {
X             ipart = 0;
X             dexp = 0;
X             rem = 0.0;
X             goto iszero;
X        }
X        bexp = fgetexp(x);
X    }
X    fsplit(x, &ipart, &rem);
Xiszero:
X    *lp = ipart;
X    *remp = rem;
X    *expp = dexp;
X}
X
Xstatic
Xfgetexp(x)
Xfloat x;
X{
```

```
X    char *p;
X    int i;
X
X    p = (char *)&x;
X    i = p[3] & 0x7f;
X    i -= 0x40;
X    return i;
X}
X
Xstatic
Xfsplit(x, vp, rp)
Xfloat x, *rp;
Xlong *vp;
X{
X    long ival;
X    float rem;
X    int bexp, neg;
X
X    ival = *(long *)&x;
X    neg = ival & 0x80;
X    ival &= ~0xff;
X
X    bexp = fgetexp(x);
X
X    if (bexp < 1) {
X         ival = 0;
X         rem = x;
X    } else {
X         ival = (unsigned long)ival >> (32-bexp);
X         if (neg)
X              ival = -ival;
X         rem = x - (float)ival;
X    }
X
X    *vp = ival;
X    *rp = rem;
X}
SHAR_EOF
chmod 0600 FP_PRT.C || echo "restore of FP_PRT.C fails"
sed 's/^X//' << 'SHAR_EOF' > ITRIG.C &&
X/* ASIN, ACOS, ATAN, and ATAN2 functions for Sozobon C.
               */
X/* Copyright = David Brooks, 1989 All Rights Reserved
          */
X
X#include <fplib.h>
X#include <errno.h>
X
X/* ARCTAN:                                        */
X/*                                          */
X/* The method is from Hart et al: "Computer Approximations".  It uses a
     */
X/* divide-and-conquer algorithm.  The argument range is divided into
```

```
     */
X/* five using the partition points tan({1,3,5,7}*pi/16), a polynomial
     */
X/* valid over +/- pi/16 is calculated, and other magic is used to
     */
X/* reposition the result.  Precision is >8 places.
     */
X
Xfloat atan(a) fstruct a;
X{   fstruct       absval;
X    register float    tx0, tsq, temp;
X    register int  part;
X    register char sign;
X
X    static float  adj[] = {0.0, 0.414213562, 1.0, 2.414213562};
X    static float  atof[] = {0.0, 0.39269908, 0.78539816,
X                        1.178097245, 1.57079633};
X
X    sign = a.sc[3] & 0x80;
X    a.sc[3] &= 0x7f;            /* get fabs(a) */
X    tx0 = a.f;
X
X/* Figure out the partition */
X
X    part = tx0<0.66817864?(tx0<0.19891237?0:1):(tx0<1.4966057?2:
X              tx0<5.0273395?3:4);
X    if (part == 4)
X         tx0 = -1.0 / tx0;
X    else if (part != 0)
X    {    temp = adj[part];
X         tx0 = (tx0 - temp) / (tx0 * temp + 1.0);
X    }
X
X/* Here's the calculation */
X
X    tsq = tx0 * tx0 + 1.67784279;
X    a.f = (0.93833093 / tsq + 0.44075154) * tx0 + atof[part];
X
X    if (a.sc[3] != 0)           /* Negate if negative */
X         a.sc[3] ^= sign;
X    return a.f;
X}
X
X/* ARCSIN and ARCCOS use the standard identities.  There's some less
     */
X/* optimal code here becausethe released Sozobon C can't properly
     */
X/* compare two negative floating numbers.
     */
X
Xfloat asin(a) float a;
X{
X    if (a > 1.0 || (a+1.0) < 0.0)
```

```
X    {    errno = EDOM;
X         return 0.0;
X    }
X
X    return atan(a / sqrt(1.0 - a*a));
X}
X
Xfloat acos(a) float a;
X{   float    temp;
X
X    if (a > 1.0 || (a+1.0) < 0.0)
X    {    errno = EDOM;
X         return 0.0;
X    }
X
X    temp = atan(sqrt(1.0 - a*a) / a);
X    if (a >= 0.0)
X         return temp;
X    else
X         return M_PI + temp;
X}
X
X/* ATAN2:                                        */
X/*                                          */
X/* Computes atan(quotient), and returns that for positive cos, else
     */
X/* extends the range depending on the sin.
     */
X
Xfloat atan2(s, c) fstruct s, c;
X{   register float     r;
X
X    if (c.sc[3] == 0)           /* Infinite argument */
X         return (s.sc[3]<0)?-M_PI_2:M_PI_2;
X
X    r = atan(s.f / c.f);
X
X    if (c.sc[3] >= 0)
X         return r;          /* Range -pi/2..pi/2 */
X    if (s.sc[3] >= 0)
X         return M_PI + r;
X    return -M_PI + r;
X}
SHAR_EOF
chmod 0600 ITRIG.C || echo "restore of ITRIG.C fails"
sed 's/^X//' << 'SHAR_EOF' > LIBM.uue &&
Xtable
X !"#$%&'()*+,-./0123456789:;<=>?
X@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_
Xbegin 644 LIBM.ARC
XM&@A,24)-+D$ \@ "&H@ $R0  ,,2Z6"H!:94   ,_\K0D9/FC(LW !( " !@z
XM0$( D P$L 4   &* ,!HJ"@ (@--$!4XL?(/'I)S& ! < %@ 1) P/@  )0Ry
XM%\L%#%AB:=E&P@Q^_X!!)  ' )-O # X\>*DS@L>HO(! O*")0(GN89">=("x
```

```
XMR#]^3P%5#%+U*U:M3[X$$R8SP)F*S.P * O6A=FLD AL[?J/WM-D#[*1Y5&Jw
XM2S9@3T?-H-7C::8,?'(]]53"$HVR],[FW?H"@&:]3SIC_LS9,U[0HNUF/EV:v
XM=.BYJEUWE@W[+EJ^\)38G7<& XP!,0$(.&,  $M _^8I(4!<P?'D(.S" P/ u
XMGS^ZGTT8'_09"U>O^$  YP5Q@%L48<?2+8N/MFN^_*C[<\<>.^LG8,R-K/ATt
XM:%4 5]UWQ5IMG:$ "#R(!4 02S7U'P)_!<;@?9V%49T-8(BS'VP 1B@8;1;Zs
XM@X$+@#P8H@,D?M9@'5_,D88;0XD"40%?C/'&'#&6%P C7] 1!HQYR0B)><Y\r
XM488<<KB!$ "(S&@&'&W4P49Y34)2P)-CM%'4D%5>"4<89)!!Y8Q?/&D&'6],q
XMV669<+"!IAE,.@G''6(,::53Y*1AEQ<0F3  >4!.J2@ P!*J*&!SHCHD 4 p
XMP $ BQ8*Z:2-#IIHHR1,>B@ E4IJ::,L:#JII99N^BD F48*:*>K7NKJJ:1*o
XMBJFHK=) JZNIBMJHK8AN.FNOG + JZBGYHKHKK<62ZRGDJHZ:@' 6FDJLZWRn
XM<&NC1"Q[;**MQNJK4=<*JVVBW@;+1+*RDIMNJ-$V&VRIHZ9[ZKGM.@MKNO2.m
XM.^VV5&B:QD %'5010PXI%-%$%2&@0$4:532 (>5E"LD!($$B$DDFH:3232\%l
XM1Y-P+/W14@LP(O //D^MTL O5-G5WGU=$+C06P' \-0RK[#B WNNQ<S6S LQk
XM (&U +QD P"VR$0 G " 4E^"%07S'GA/63,,(V2YS#.%IMWF%3V_E0,$$L! j
XM0]Z0$(! %01/AZ-*"S]@]A0\-LP!A'M<X^TU@+JU= 8!-K44'8!');6B$D.\i
XM]0\=88]=]MD$J-T"%"6EMV!9\&R-UFQ<8Z[Y9J]-?3)U_W@P]$+R^<7#+Y  h
XM8$S6)_<LLUL(G"Z ?-(\!<HW]PSQ.6@^%UB[M0/(!\OI!,B'RH:69PV@:U[,g
XM;H8*_>75_(-Z@Q[\S [\] \G0Q6UXH/8=P[U@MD30&+YFP-@83^Z@.'+2%]=f
XM;Q7TTE-OK?7G.Q]@^[+[F5NZ!Q3PY45\3'&*5<AW/_,I:$( /$V*N/8^$\#'e
XM!9!C'^BX,#OB1.<K8.!$A+0A!/L,!0R*H!\_^O8 ,V"@;PM@PX%X )AL2"V!d
XM"*IA$,#@%Y8P0(.H<1\ ^($+[\"'A6TPP ?CPPP8]F2$);0+/SX#!EJ,4!@Fc
XMS L8A$0 :DP@&3]XH"8$D0<A# 4PU_@!8+ !!$LL8QQ!T.$7PN C( U 2#0*b
XM@XOL*"3SV&*.-L+1D/HH "7,L8XRJ$@?!R !+&GI3E[2$Y\&T*4G16E*?<+3a
XME\($R2?1R4Z9I-&1DK2D-<TA#G*@PYW^I*Y-N5)1L&05KM#UK7+]:ESELA>Uz
XMWK6N<<G2EN("5J.LY2QA+H26O*QE+XU)3%T%TYG-W%>\I%FO<#'$6;_<);*8y
XMB<Q39<N8[)IF*U^E350A4YK9;)6QQ"G-<MWK6^M\I2S3&:QH4LJ<U214H_)5x
XMS5,-:UO>#%>_L F <!:3G/ \9S(7>BI[JE.AC (7L-J9KG_>,Z#&'.@]#2K.w
XM=\XRG]QB:+KLF<N0RA*@[@(F1U]94GC!BY[#="8PF]FIFI;'IA'-Z0!L"H!3v
XMIE)@"B'80PZ&$0$<+2,;&9(Z(+( (X1$A2\8  S-@("BF. #*UI)2XP& ,2Qu
XM@0'>,R!1,B(.2\@$"+\)CELJ@HH.R.0'+9&"[WA B%^P(VX\, 0J0C&84- #t
XM'[\ 8FLZU\ (>JU^4A3L:[((&OA,P2[_6$]A09<:VX ./K\1!Q*F (1H0':)s
XMI(.!6P$ A):03@LM0J4J!TFF42JI(I6$DI0ZN4DQA3)/>UHE2]TUT9"VJU$,r
XM@.@NI751>1GW6Q0(UZ.JV:CD&G.Y'2UG/%T%78HFE+F.TA<NAPNMBL#A#7<0q
XMV$.$:C")8"0 Z&!84@?@"J9NZ6)?.4E*A$"$J;Z0)0]P@T\*&#ZC($4I"80Ap
XM&Z((  9DQSCM_>$ /C-!M%BA@P@(ZU#$ XRSF0<'( . R!;PGQ].5GV 2)\1o
XMO;)"(IR! ;KYQS^(8YR9J!BT %C<%[Y[AQQ9R4A(>FV<;FPF-*F)3$]RTQO@n
XM%-LL;8F29&+#&]Z2E]A>$I)&PL.6"% E5K8TI5@&KG9WVRKGLC-8U0W6=%VZm
XMTP++-*7=5:DS+5JM+8>4N)8"0!FD+-Z@-F2HYJU( 4*%5(Y (&(0.0 JGDH2l
XML S@*>T%@!$4&ST!GH$"WA-*7D"0@Z%<=7S[(P"B*[+H#P\6+=MSRUC!P(RGk
XMX$(5.R""8KO2H0][YVA>%< 4?&<5) PCJ@P.\7U8#:'^/246<NA"+YY2#V;Tj
XM  =/:<<;MC +Q7*N?<[N&F4Y]+]IBXX?OS%'9R&;5ID<0&QD,UMY(K<VQFY%i
XM/OJ JE5"!(P+\D)!+:M?_VAC[M"=9D7J!E"(_.!N>*]GWH2U;!!5E,!\(R!$h
XM/+B@*Q K\+V0V'X-7ZROI9B^%?UC2']XX"(J$8P@8.(&[PA"*?QAC2#4@@#Rg
XM"$(QSH""(%R#"/L(@CJ \;HH3QF/+7H1&LK3QP#8H$8WVCEKAQ0 //3H1T*_f
XMHXYTX<@CK\FUI903F&R+9$T^.91L.E.:=DRC(+^)ZYZL$VTEJ5N3CO.6!XUHe
XM.!MUS7!YU)_A&C-OB_LMDI;SF[<Z:*OX&2R-*K-;QQ5IFXW)9E<Y-%B%;VBXd
XM+ HO+CNS[02UNW412OE&<0&9C?+[/0_OSG("\U27QV[A8]K/X9I^\.*$Z3-1c
XM^BW-MRKTQ<5IF74J>YXJV2!+LG/!_(2PAHA#O0,#Q;@#[0="]Z5ZFO9T6CJXb
XMW^]-N-)YP?3)4Z[84+\% -AXRLGAT;+GW:?1!:H(+UH@D[P<_WPEG+CW#UN7a
XMB/,%95JC^*X?GECY'Q9E4!M ^E6QB6U0A0?4( D?$#<NDSZE\3N?-FWU5EFKz
XM<4(Q]D,\ &P5\ -:Y1* , Y T#HR$0AD\VQYT6"7]36C55H<Q@/;UWW5UEA4y
XM8X($P'VK1F*DPP',PP/>L 750(#Z5AUN (($8'%Y\06W9V,T<GLQ  .*I"."x
XMT'10!G6P)2>?1%M3-W:Y=5NRA4E5UW5M\G55]G>NXF5I5V;0Q79N5TY>&%V w
```

```
XMAUS=)"_=-4]GAT]F&"P<U7FULEUUYTN5UX:'!RJ8!P!Y6%![R'G+1'F-YUO!v
XM@G>H9RIKV(9Q&'C*HDM<6%*D=T\ 8 9M\ 9D %3'M'M$Y3"R 'P+47R0L  5u
XM U_\4($=(Q-_  LR@05O@ "#<Q4)Q$#K]X&Z-A0*%W$/YF@:X$04 '%3=!KPt
XM 5JNH$*F"!.HJ(H P(JN>#^QN$ ?QH.W^(L.UD&[>!-LT(L3)W#!*$5@,(R%s
XM5HS!D8JK^ 8>\(H@ ",,  * <(X%]@)L\V'C\XRUF!?> 0;00#_PUX[*!Q\Kr
XMH6*GJ&'(R(H&@$ .8G]Y\7XND&"_PX,'^&%@@ PU4@9I8(4X9P9*]@9R<(1Yq
XMH09?4(ED "=#UT5EXI$\IR,<D'4^QG5L(F1$)B=&!H6<1(5/2(5D1X575W56p
XMMDO41(C M$YZB$ZK]V4OA8=ON)-75GH/M2WA!$R)AV6LYRIX=Y1OII/CY'C8o
XMLH<^Z8=(Z2I\)Y6#2%"Y(H=PZ$RN!Y608@9A( 9S@(GDQ7OG10O -R2PP%08n
XM9C$;\HH5=G$#\ ?X9@7\,0#@B(ID(P,^A 3!  V!H!U*, AN4%4 <&D)9 L6m
XM]@=O8 &1-B,+AW$ \HI@P V!.1-D PB#T $DR)G$L"%3Y0#LR !"  6OF%4<l
XM8XP:IH&DM9HP  B?"0B <%:D:5K04"9I*4A*=V-L0 9S5A0 P$424"9R<)P:k
XM210X1DI-&!% .8AD]H6]E9UT%R^-$H:P=)U?9I14"9[86953N2]T-&1L>6?Ej
XMA3!#@@MQ.0 L,20$$&AU28I(X ;RYA\?MD1]61$$UQ0E<'V'-&0E>6,M,@8_i
XM0@9UP$4@T")SH*!NT)((&J$_0J$Y^2UM=RI>9IWI$F8'1!!N0 =F(#"E8!%5h
XMD"V;"  &X 6>* #PP%2@!%^R<!(>P $F ",0@ (P0@'TI003\!8"P ,,, $ g
XM4 ),HPI"4 1"X#*C!0+_D \]P =ZJ0]"H)_\,*5ZZ0^CI0#_< ]+X [_< M"f
XM8 1", 12L A% 0$3(&Z3!@1"  00  '!,1PX$*=L2ILP( /6H@##, S2  0Ve
XM$ !*@*7$T0-#\ _^  ;*@#AF  $Y 0!0F@]4-5HP$*55A*C_H ]@  MX.@'(d
XMR -N@HP < 9J@(P@< 9$@(PJ< 9&@(PK< 8X@(PM< 8J@(PN< 8:L)H/PJ8Oc
XM 5=.T#I<\ 120 1I2CHR,*4+ 4(QY@4Z8"T!0#I6T %3,#HQY@0VP &%^A5Gb
XM@ ":NJ@QY@.:RJDQ9@-3L B=BHQ>,"5(@XQ%<'V$@(Q'< 9\$)!F0 FC!0%?a
XMVH_W\*MST0!*8 1GP 4@T(O21@#>P \ H  !^S=)P X64:0 @ =F\$,-\ (-z
XMX!H_8!?^L+&;VH_YX*_]B _^^@,%@+$=!D0)"P 6P$:DHP-% 0,<  :@T )4y
XM\ ^R\+ 6@;$>ZP^^"@AP51:R\!DKRP!:.B2+R@G(* 9M4!WN@(QX "/^8 ZTx
XM*09D4P[KN <(T(/0P!&6P!'(P!%BLA"J,#!E2[8< ;'",;8"H+9MRQ&7)QQNw
XMZ[;")QQG^[9RRQ'DD+8<00Q\F[<+00/RP0@P0 !'@0-XL"(\ *W4T0_F\ -]v
XMBK$-*P(%ZX$(J[ ,*[ (H+/)4QWJH+.W4QWB  ,40#9@H VC)0!?*A^<,%HFu
XM<P_R(0F+NQ"6&J7*:@";JJPF Z[^( ?R00<@P 14P @:,&@!$ 3ML!!"  %-t
XM>C)/BJG5X0@J0 6<"P;JT+C:\ 4S)J(DBG,U0@=Y  =EH))2@B86HI)S,!!Ls
XM< <9N2;;2Z+%N29/LKTOLB9"9B&9E*'CY*'B*2AIEF:>\K^OU%M:UEM?:9[Er
XM&2W=Y;]Z)Y1.Z< )C,!YUU$ /'GDN;]4:<'CB68#/,%=%IX2W"H@>GINYHABq
XM!L%IV,#?LI0!;,)B25 =T2@Q+!S!,L,V7,,X+,,Y3,,ZW,,\_,,W[,-!#,0Xp
XM['@N/'>$!\)B><!PUL+[*\#?F<'F22H5_&4]):$EBA"$<*($D*)N61$28"M]o
XM!B#S.0"8$&B,8'Q(<!(?@*,EPZ, X*-'H 048 8*$*>'VQXXA(XOL #!FHQ/n
XM@&%*0 1+(@Y7JJ@,D%Q(NA#&(!?/JF+*&@ GHZQ>:J5&$,F*BLGV<+0# !;[m
XM,P"MDY__( >RD6FB' :N$000&P!" P J8,>'[ ^'; ]2P BUO*84X*;DAJ=Tl
XM*A/# 0*Z0*7_8 8"L*1X2@%ZRJ<+.PS%$*@Z$  ]4 2C@PQ38,N,@,NZK#:\k
XM'!S,$33'K "E!0*AR@9F0*H#RZIGH 3'C M < 8B,,A?I01'0%4PP ,J)A_Aj
XMH:-]_ ,]**Q/  ,'(!_FP*M^W#K>(<CZZ0]_@P7MA0&U3,=PX@_<(#^Q/,NDi
XM@P?5# :F@(Q;T :@ +5NH FTN057F[5;ZP1="RE.D# JP-(N#2 M#=,OK3X5h
XM80!G;!'G8A$[H=,500 [+0!0P!']M-$+0!(\-(($-,$@&%$G=0O?0 *2]0$g
XM,-1.+=5"0#I(<,SXC,V0H\URVLLJ\1;^( ](0 P(,A300 A>/:?! 0&*DS9\f
XM7-" C&&XT!9L8 WRP0RS&ZW5@0CR00B'K ]E?=9YD=9K#=8  0]E?=9YD=9Ke
XM#=80SE@ X08 @Q<+G$ "@.4)#@"X^  (P 8;& #^%L(! ($"-%X (!$  40(d
XM$ @0@$DD<(8 #QX3E;CYI\_,1"T])"J@*60FOX@ 2M@,P" G  $T"1I$J)"Ac
XM0X@2 [3!*"2(NW]A8+B@B018-&!"(" A!@*F2&B$2)I$J1( @C,  !B#04 !b
XM$  @?@! ,C.,DUPA"7!YH@*($B)G$.SB%]B, 25%;/:CMSCP&30P"DPIH@3(a
XM&3"'SW#1RA78"P O!OP-#.6)5:PB&DA"!"V0";P%#R9<V/#AV+*!T:H]F5+Bz
XM6PQT[>+5RQ?K:I&#"X.A1!>,.IG__,'MITHI!A50&!0,\(^?8Q #B "ZK<1(y
XM8A6#I!RYV$^1CD%@[%4'\,\'4!AFV*"4;DWU!M5$;=@"%!YG4  4%F<X8!0"x
SHAR_EOF
echo "End of part 1, continue with part 2"
```

```
echo "2" > s2_seq_.tmp
exit 0


From 7305245311458054135
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,125cc2d0df676eab,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-06-03 12:10:18 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
agate!saturn!koreth@ssyx.ucsc.edu
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i042: fplib20 -- New floating-point library for Sozobon C  part02/02
Keywords: shar, C
Message-ID: <7810@saturn.ucsc.edu>
Date: 3 Jun 89 19:10:18 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 518
Approved: koreth@ssyx.ucsc.edu

Submitted-by: dbrooks@osf.org (David Brooks)
Posting-number: Volume 2, Issue 42
Archive-name: fplib20/part02

#!/bin/sh
# this is part 2 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file LIBM.uue continued
#
CurArch=2
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> LIBM.uue
XM/&V5QD7_L&$4!#1=Y(\PV)F'D3/B ;!331@1 U0+5P#35@ ] %&>3?YX M0*w
XM-*KBH21&(:6/AR),V"%&&&PXI#\.]%2>AP9X*,!O9A$@7$G$M1700+DQQ=M3v
XM0%%D$0#\:,211R")-!Q;*[7TTD1*.J$D/TI*80< 0KCR#Q9U_D.%DE"$%A<Fu
XMF6W6&5P ./+?5 '( 50.XZFA)!.D,9$.$%U]%!A)I#DQ#J5> 0,#!+!$!5< t
XM,(1ZU*@6F#H H>Z8RE)<OIA: *&2F&K &2Y$.FFE90X !*9;:<KI5SU \8\3s
XM VKIU$,8FI'IIKR&-  @)"D!!5S[J(*==@#L \T/6T'QPE94C'MG:N#^(T6Zr
XM_#Q'@#>,2=""C$AX!UY[9RB@@IU(R'?1/ICL^P\28-#S@ED#*& N%>ZV)@,!q
XMDD@$ Q+!U#:OGAB6B[#"6V'1\!,/1PS Q!4'<C$64LP$!88L'"RMPJH!)I+#p
XM$$M,<6VW8<C BBV^&*.Z'NK#LXM1_<R/A^@ Y8(9"AA%'A,>&BK1"F<8L"+3o
XM1@WP#]08X0%E<&E1B28 +"%7UUT@F%H&C<STN+6'2E2GC@Y;X:.D/4)L)8>]n
XM?19Q!@X_$-"<'.X.=C$77;8A ;Y^*&$G%SU'-:@#V-D#ESXT"%'!1?JXL.T9m
XM FA 5I:[+7N@5 8(&ID%GP. >>6@:^ZOZR8$/GCA3QR.+P2._P,YT3%Y9@H8l
```

```
XMQ)#F1<'&@V'($E?)(04C#%  @!<V :#)\V[3$ST 752]?0N.:<"-DMI+W[T<k
XM2E  5SZ2D$:/J0"<(86I 9P!JD0"M"0^&-J0GQ\(#:@"M<0&@9_)8RT4D,%6j
XMY"$+(+A!!%5H! (5^ \& H$-#""-/%)600T  PSHR!X8=/&]R$WD9_0 @RHTi
XMYR%9K"D &"+#U\X2MK44AR-P\8<DDE4Z W6)#=:X"#[4(41O &  1TPB <@&h
XM@ (T\8F0.( ! ##%*E+QBE.$Q "RN$4L3L2+5A2 %\5(1B8N\0  0",:D8C$g
XM)4+B 4[48AP/$("0""+":BC%O58@  D0 $+8$ #0!"7+Y@!#E\@PQOB0HJ0f
XM%,"0B'3#'!@9&"! \@MSL$,8V-!(D0#B"Y@<0QC<8!-1.!*3=)##&."0!P @e
XMXI1?<$,=VC"'5+X2$H_\PACHD <XE,&5IZRE*M$@!V#BD@UMJ ,; G/+1QXRd
XME*.\I0'0J$5J#L":V$RC-M4H+6YFTYO;[*8XJRG.;.*2F@7@YCD=B4XT.K&.c
XMV70B"<+I1!9HDYSR#"<YR6G.:X:3FTZDP3_OZ<]O#G2?!=5G0L'I3@ (E*'Cb
XM7&A$U9E0?@X4HOTT*$01BM%R>M2@'/VH1RUJT7@>Y8EB=&)*3ZI2E+JTI3!Ea
XMJ4Q72M.7SM2F-8UI3F^J4YRZ5*,B#>A%)ZI-H78TI"55*$4S2M2&/G2H))7Hz
XM4@D*3J,>M*+^1&HZB;K/K29UJDB5:E$=.E2F@M2?3H1)1^,H47R*]*AB=2(1y
XME I5L0*4K' %9U0;JM:R<K6B6^VH7L4:4J!.]:SJ]&I6LZI8IJZSL']U+&/-x
XMV5AV1M:C5MTH5O-*57KB]:I@72QBF_K6RX85J.^DJV8ERUK3;M:S\.SL8 W;w
XMV:2F=K:N':U;F5!;T8*VMW<-J5Q_&]J&"A< <\5M<<7)UKLF5[5TW>ME6TO2v
XMRDJWG9C][&JCZ]O'MK6[OMUN6KF[7:!"%K*T3>]TH>O6OIKWM<H=Z6NK.]KEu
XM+O>ZI?TJ<65;5]PF=9VZ_:Y9G_C4[:*7O5G5Q#/A( <ZN&"1=41B D)B@ #8t
XM(BX/0$5<P*"!N!R@F%IT@+00QHB0*, )5H@+$L[! 0!8Q2TF@ !Z@,&+?V@1s
XM "B(GP-FW*L_Z   U5-$"R2"2P2,P!T<"8P06JE% FC%(*;RFQ=,Y9Y22>0(r
XM9T""VLY  U.9X0P>,-49$L.$;P  !T[P@A/J "X .&!<;@G#4= P&!"V^<TNq
XMB/-$:%!G;-P9S@@( Q3JK(P9,$:/ R  &)!AZ"\V&0S>4 (A%OD//JC.# P8p
XMLB,1@"$UC(,0DG8#.I!0#! 0 !"%$,$ A-$K0  B$"]&  @ @2$EH#@[")&)o
XM098) %T8Y0#9$<(9TN4/0,M9 $QHC0F83  E,'D .&B)!1C3 @OD0UH&F#8 n
XM7% B\OCC#1!P6369(C,"M(;<@6E(5ZX!#&=+BR52@ !,BF !8$SAV3IPMQ9Um
XMP 8):+L%,,B!M'1 !"#, !@#9T K%3!P-A! VS#X@<#W+>Z$H5LD\#*1";Z0l
XMYC7+&P!E  ,KGLP *0#A9_[ D!!NK8]<YV:9 9"01 P0[#.0QA\%UXNQ)T('k
XM93-;W]"6-K45[N@!\#LNL("XKAG@F,\]G#&>FXD_S(  (4Q@XQVO ]#Y38: j
XM#[S@1B,-/V A]3<8H =2+X^2]$$:?9"]/&Q0 ,0Y^"-VB#T24M_6&SQ [(H;i
XMX.+O8HP";KYNL$P #([P^KZCI44#D 0)XDI(N5N#(3A(?.!^!WS&!S]QHQ<>h
XM0S) NT&RXR%YW-H?2V?#V6=ND&[O>O5?9, /8!* V4_$V)!_@L+_T8^!MR&#g
XMG=<!V0WR!@J0? I T'F> PV ?M@BZR1?"+Y1WO?ER]D?;  #.&ZM#=$SH(+Bf
XM)@"@/]:%8 A#5!.QA]-PO7QW-61>;FF./9KCCN;@8^<"H$+&PUVWS"/, .Z2e
XM<0?Q?^[W!&&W%?P #3.A?C)B#TOG!A) <F!W%QFT%?80#;GA>@P <S@1%09!d
XM<B9'0?S00&R@!26" G G!>%G+N[P,5=@?N@' (( @D @@B2(' C(0?S >%M$c
XM$A  $C# /V+'@XY7$A!P9VW& '(&)@TQ*"2 '?+@!AH0?B1'A*?A+AQD#S-Pb
XM%?X@+3BPA?]@#P.7=2TP$TB& ,U!#_1G?SL7 "2P?^9R?PCS)#.A#0'(& S0a
XM''98;@VA@/] #RCD/AQ$#X."@^6A@T3H@T (!MR@AT,(#);2>(\'!1_C=P5Hz
XM-!X""R=(?!AA"D:18PS@(9R@)"F'$1S <FOW9&@(B;U2A$, #2V0!.5!$.7!y
XM !   #70!D]8$ I@!AXP!7;"#PS@"MA6'F4@(#X1;[0HC+>8 VR @_''BMBFx
XM%A#0=EFWC#\X,L"X), P 8P1 ]BV?C^B#+=F,-:'= V!<F]2(A^0'6^  @<Sw
XM"!+P!C<P?I.7.W>! %;@$\;&!^DH(RD'#26R$]C'!0>C#W&A!/;(&OCH%E?@v
XM$^+Q#^@ =RR0)W;C$YHF$G\X("!@)_B@%#)F)_2@%,87C%EG;!CPCZ3'"6G(u
XMAM87A,; <K'6 D@PBSX1!.AP1"B!!(#0%G_P,_I   4A(VRW%?J0=>4XD\;"t
XM#R" @#0F+8" 0@)@)S=6'L:F"BII#]T6AFS@ 7G"#V7H%O_PD;G1D3;)$6:)s
XM#] '$R"P$ O8%2* @."0"BT0!.6!'?20+TU9'NIQ,?Q@+NU2;BX " LI$A>#r
XM#R(' T'!!F2PF(U9 HT) )?$8&G@!J:$2ZBD2JQD3(\D3&Q0!@713)A$F6Y q
XM!V8@FL+D!JN4!\UT2&/0!ANA1:T)!VY0!G AFXYT2,FT3+B)F8?$!G3P!C8Qp
XMFW-0!V(@+=+46L$55_)E7(KE1$C44-%)6OA5G? %7")U6N1U5="I4-T97\%5o
XM68(U5M898-BEG$\TG=_96>O957<UG0W%  /E1-(#4>MI4.MYG\R51.])4/KIn
XM5NKY17?58K15GM3978&5G>&5GF,%GQ'5GN!U6*1EG_S9H-M)7^MU7A'*GA7:m
XM4/,T8%75H0Q*H?(Y5?DIHOED4@Y*6>CY6 GJ72 J+?H)H1(:HP"*G1J*7<VUl
XMG_;D7Y.UGP666T)Z6R/Z1 2ZGP%:GSBZH>8Y3M8EGMF9H#OJ5CUZH=<Y8/]Yk
```

```
XMG?J)GA)*I!8UHP;16Q"ZI9:EG6:*5@*ZGT=Z7$I*70*670XJ75\JHO'I64EZj
XMH4_JIEA*IQGZ6MUIH/A%IO_%I#):H0#6IRQ*GKXU7O?UIN<)18L* .[%52\:i
XMJ3 JI(,:7]K9J#:*H._56E.*II<*5I6*76?JHQ*:H]@Y7"8* +QE3@[P3',@h
XM2F[P8'%! %4P5Y!081=V%+*P81TV$3D&"0M@!"96CG#&  7HET,&"'%AE_CPg
XM9!@P!06! 85X,,S #,W0 W'X,8?)B' & >XRF-UZCZ<!F-XJ(_S@. "@/G%Af
XM",WZK&.I:PI *)PP9-M@#OW0CMI#C%K$<!30!AY@+N8XENDZE@3@KPG3KF" e
XM#?1Z!EHP9-FZK1N99Q0 C.VJ!( @$ 1;K@SYK=X0A^AZCWZI:V^!9^7A+GZYd
XME^VB9G4P9* 0%T3PAV! "D-6"0*0#V) L_Q!!I<TJZ-TF<X$!\ IG)RIF\J$c
XMG+D)!V% !HW9FP5P2+5YFP.0G$3E1"7*H9=J6U?;K@A6IZW:IAQZI*!:II:Zb
XM57%4JOLIMFYUJ'.ZM6_J5FP[7V@ZMZ.5HA,*15)J64 &!Z\)![;Z11)&818Fa
XM+8T$ !PF+<.Z  \!"0H  J=! ")P&@@@/<00 &I@ /,$ @$@ @( "P%@!I\;z
XM"&L&N633$@P' ("P9JX)F](BM)#TMW$AM%)KFZ\;&!P 25,[N[B+G>-97MWDy
XMFF_@!G9@JZ5 -KE*N+TJ *^$N,$: !U&K(U[8E9 $^R* &R@ D8!"-G1E&[1x
XME_EH;!I 0?Y ,:]V,?JPET?ILJ2H?2BF8N>  3BV?$RS$64& !B0=38 $Z Qw
XM:0;P:5,P!/]# #1F8TB$!,, "/C!#;?VEP-@ LO'8U!Y!BJ@ (, E:D1!C[Pv
XM?NK:._R 067AK%ID A]PC0BH'F @#QFL 1Q\E\'(!K+& W]@P/S#LF! #>^[u
XM%_&+%\M7P#=V&]Z+ $^I17_P$$ !!&W0;(-0OZYJ9OGKLD !!F]  '#P#V#@t
XM#I@@$4   VRY$$30%>"P"NC!B)!4M*5T2H=4!\%YQB*A Y!D!L&Y3)<IE&\<s
XMG'4  '.L#+YKMO:EH)^ZI.!96B&J7X&:782\H#L:R*BJJ*FZH8F\G_@5R(><r
XM7H=$!FE0O(LDN  P8;M:N!.!!,"JM,3:8HXK D.P$9-[NG!  I1[!FX@! $@q
XM! )@ CP@1EHD"X)0!E(R")(@".*@".*P"&8@#^V "$H0"*!#!8 @!W]  @$@p
XM#*=Q ,1P&@L@"4(@#1(1!!J[!@0P!4$ #8 0B=-"!@0@#@LA!=H,9 3,!UVHo
XM!$'P!B(@"Q+Q!VY@ 1 0 !(0S11 S;7PS##A:@%@ @FP9E7,S&! #AN1PI5\n
XMR;?KFW!@R7."QQ2FJ<RI7[_[6Y'L61;-7W^<T1AU2$U+!H$;89NLO'$1 +@0m
XMRK<<$HQK8@JCN@"P *9, D1@NI)+N3EF!W\@/;,@ &I !<"P:@,P3[\JNK( l
XM"&\@:YTK -)C! \' )80RW1B83"1 6[@ >:0"@N!S""P!@@P!0+P!B  SFJ k
XM#^(,"$J@SBK1A1 @ ,CPSX!   *- B=6!PEM#T$M#$,- $4M $=M=ISKN=*Sj
XM$ %@ 70RU;90U6S #%E=&V1@ .8, %(@ &O@ &=-!K 0V9.M!K)0Q73P!R)Wi
XM2,5YG%H$NZ)MG+R+2R#MM U-  8 22&=VJY]3PD:HQ=]V_N%VX]<7WZ%V^.Yh
XMT8-L5]N$M&PPTDG$R;QZTH:@T@,0L\2J B8& D,@ D1  D5@TZF, )]M @) g
XM#)Z!!\80!&? !]0P!)2@Q4(PU38@%\& !$/P$#H0 $A *4&@#," #,& #8.@f
XM#81 !@A@ P6@ [0'!@0 X$)0!$A0!,!0,85 SIHM!-P,SE,@!)==SN<L!&E-e
XM=4HP!&:PSNTL!&)-"_-\U6V=SVFTSS'=T\0 UP)0P2B0 ":   2-N/3PV55,d
XIQL1-!Q+MT+N9VE$+!SU>VA.-T<*MR+U]Y,!]Y,95Y-#5T5>JY($!&@#Pc
X b
Xend
SHAR_EOF
chmod 0600 LIBM.uue || echo "restore of LIBM.uue fails"
sed 's/^X//' << 'SHAR_EOF' > LOG.C &&
X/* LOG functions for Sozobon C.                         */
X/* Copyright = David Brooks, 1989 All Rights Reserved
         */
X
X#include <fplib.h>
X#include <errno.h>
X
X/* LOG:                                          */
X/*                                        */
X/* This yanks out the exponent, performs a polynomial valid over +/-
     */
X/* sqrt(2), and adds the scaled exponent back in.  Domain error on
     */
X/* arguments <= 0.                                */
```

```
X
Xfloat log(a) fstruct a;
X{   register float     t1, t2, t3;
X    static fstruct nhuge = {0xFFFFFFFFL};/* Largest negative
     */
X
X    if (a.f <= 0.0)
X    {    errno = EDOM;          /* Impossible really   */
X         return nhuge.f;
X    }
X
X    if (a.f < M_SQRT2 && a.f > M_SQRT1_2)
X    {    t3 = 0.0;/* arg between sqrt(2) and sqrt(2)/2 */
X         t2 = (t1 = a.f - 1.0) + 2.0;
X    }
X    else
X    {    t3 = (float)(((int)a.sc[3] << 1) - 0x81) * 0.34657359;
X              /* ...(log2(a) * 2 + 1) * ln(2)/2 */
X         a.sc[3] = BIAS;             /* Scale back to range */
X         t2 = a.f + M_SQRT1_2;
X         t1 = a.f - M_SQRT1_2;
X    }
X
X    t1 /= t2;
X    t2 = t1 * t1;
X
X    return t3 + t1 * (2.0000008 + t2 * (0.66644078 + t2 * 0.41517739));
X}
X
X/* LOG10 is easy. */
X
Xfloat log10(a) float a;
X{
X    return M_LOG10E * log(a);
X}
SHAR_EOF
chmod 0600 LOG.C || echo "restore of LOG.C fails"
sed 's/^X//' << 'SHAR_EOF' > MATH.H &&
X/*
X *  MATH.H        Math function declarations
X */
X
X#ifndef MATH_H
X#define  MATH_H
X
Xextern   float    sin(), cos(), tan();
Xextern   float    asin(), acos(), atan(), atan2();
Xextern   float    exp(), sinh(), cosh(), tanh();
Xextern   float    log(), log10(), pow();
Xextern   float    sqrt();
Xextern   float    ceil(), floor(), fabs();
Xextern   float    ldexp(), frexp(), modf(), fmod();
Xextern   float    atof();
```

```
X
X/* Some useful constants, generally to 8-9 digits */
X
X#define M_E  2.71828183
X#define M_LOG2E   1.44269504
X#define M_LOG10E  0.43429448
X#define M_LN2     0.69314718
X#define M_LN10    2.30258509
X#define M_PI 3.14159265
X#define M_PI_2    1.570796327
X#define M_PI_4    0.785398163
X#define M_1_PI    0.318309886
X#define M_2_PI    0.636619772
X#define M_2_SQRTPI    1.128379167
X#define M_SQRT2   1.41421356
X#define M_SQRT1_2 0.70710678
X/* This is what the compiler should turn into 0xFFFFFF7F... */
X#define MAXFLOAT  ((float)9.2233715e+18)
X#define HUGE MAXFLOAT
X
X#define _ABS(x)   ((x) < 0 ? -(x) : (x))
X
X#endif MATH_H
SHAR_EOF
chmod 0600 MATH.H || echo "restore of MATH.H fails"
sed 's/^X//' << 'SHAR_EOF' > POW.C &&
X/* POW function for Sozobon C.                          */
X/* Copyright = David Brooks, 1989 All Rights Reserved
         */
X/*                                          */
X/* This uses exp and log in the obvious way.  A domain error occurs if
     */
X/* a=0 and p<=0, or a<0 and p is not integral.
         */
X
X#include <fplib.h>
X#include <errno.h>
X
Xfloat pow(a, p)
Xfstruct a, p;
X{   fstruct       r;
X    register long pint;
X    register int  sign = 0;
X
X    if (a.sc[3] == 0 && p.sc[3] <= 0)
X    {    errno = EDOM;
X         return 0.0;
X    }
X
X    if (a.sc[3] < 0)       /* ...if a.f < 0 */
X    {    pint = (long)p.f;
X         if ((float)pint != p.f) /* ... if nonintegral */
X              errno = EDOM;
```

```
X         sign = pint & 1;   /* ... carry on regardless */
X         a.sc[3] &= EXP_MASK;
X    }
X
X    r.f = exp(log(a.f) * p.f);
X    if (sign && (r.sc[3] != 0))
X         r.sc[3] |= 0x80;
X    return r.f;
X}
SHAR_EOF
chmod 0600 POW.C || echo "restore of POW.C fails"
sed 's/^X//' << 'SHAR_EOF' > README &&
X         FLOATING POINT LIBRARY FOR SOZOBON C
X
X          Rev 2.0 -- prepared April 30, 1989
X
XThe files contained in this archive are a set of floating point
Xfunctions for Sozobon C.  We provide:
X
X    sqrt      exp       floor
X    sin       sinh      modf
X    cos       cosh      fmod
X    tan       tanh      fabs
X    atan      log       ldexp
X    atan2          log10          frexp
X    asin      pow       atof
X    acos      ceil      math.h
X
Xand modified versions of:
X
X    fpcmp          fpadd          fpmul
X    fpdiv          fp_prt
X
XThese are available in source.  There is also a new LIBM.A that has the
Xnew routines included (and the other routines re-arranged to remove the
Xneed to scan the library twice).
X
XThe complete archive also contains design notes, release notes for
Xrelease 2.0, and a brief set of specifications.
X
XOnly LIBM.A and MATH.H are needed for normal use of the library.
XLIBM.A should go in your LIB directory, and MATH.H in your INCLUDE
Xdirectory.
X
XThese routines may be used without restriction, but I retain the copyright
Xon them, and the Only True Upgrades will be those I make.  If you have any
Ximprovements, please send them to me.
X
XDavid W. Brooks
X36 Elm Street
XMedfield, MA 02052, USA
X
Xdbrooks@osf.org
```

```
Xuunet!osf.org!brooks
SHAR_EOF
chmod 0600 README || echo "restore of README fails"
sed 's/^X//' << 'SHAR_EOF' > RNOTES &&
X        Release notes for FPLIB Release 2.0
X        ===================================
X
XThis release contains fixes to FPLIB release 1.0, and fixes to the
Xstandard Sozobon release 1.0 floating support.
X
XIn the standard support:
X
X- The original floating point add/subtract, multiply, and divide have
X  been rewritten for better speed and accuracy.
X
X- Bugs in the floating version of printf have been fixed.
X
XThe remaining information is of interest only if you have been using
XFPLIB release 1.0.  The following changes have been made:
X
X- MATH.H has been expanded to provide symbolic definitions of some
X  common constants.
X
X- A few minor bugs have been fixed:
X
X  * Some routines could return -0.0; the sign has been corrected.
X
X  * Some routines could return a number with a biased exponent of
X    zero; they now return true 0.0.
X
X  * Out of range results are now signaled by ERANGE, not EDOM.
X
X  * frexp() returned an incorrect value on overflow, and ldexp()
X    didn't handle overflow or underflow correctly.
SHAR_EOF
chmod 0600 RNOTES || echo "restore of RNOTES fails"
sed 's/^X//' << 'SHAR_EOF' > SPECS &&
XFPLIB implements the set of routines described in K&R, Second Edition,
Xpp 250-251.  In the following descriptions, x and y are of type float,
Xand n is of type int.  All functions (including ceil and floor) return
Xa value of type float.
X
XRemember to declare any routine you use, either explicitly or by
Xincluding MATH.H.  If you don't, the compiler will assume the function
Xreturns a value of type int, and will perform an unwanted conversion.
X
XA domain error occurs if an argument is outside the range for which
Xthe function is defined (e.g. sqrt(-1.0)).  In this case, errno is
Xset to EDOM and the result is usually 0.0.  A range error occurs if
Xthe result cannot be represented as a float (e.g. exp(100.0)).  In
Xthis case, errno is set to ERANGE and the result is HUGE_VAL.  If the
Xfunction succeeds, errno is not zeroed.
X
```

```
XIn all trigonometric functions, angles are expressed in radians.  Pi
Xradians equal 180 degrees.
X
XAll routines work in all three resolutions :-)
X
Xsin(x)       sine of x.
Xcos(x)       cosine of x.
Xtan(x)       tangent of x.
Xasin(x)      inverse sine of x in the range -pi/2 to pi/2.
Xacos(x)      inverse cosine of x in the range 0 to pi.
Xatan(x)      inverse tangent of x in the range -pi/2 to pi/2.
Xatan2(x,y)   inverse tangent of x/y in the range -pi to pi.  The
X         angle is chosen so that x has the sign of the sine and
X         y has the sign of the cosine.
Xsinh(x)      hyperbolic sine of x.
Xcosh(x)      hyperbolic cosine of x.
Xtanh(x)      hyperbolic tangent of x.
Xexp(x)       e to the power of x.
Xlog(x)       logarithm, base e, of x: x > 0.
Xlog10(x) logarithm, base 10, of x: x > 0.
Xpow(x,y) x to the yth power.  A domain error occurs if x=0 and
X         y<=0, or if x<0 and y is not an integer.
Xsqrt(x)      square root of x: x>=0.
Xceil(x)      smallest whole number not less than x.
Xfloor(x) largest whole number not greater than x.
Xfabs(x)      absolute value of x.
Xldexp(x,n)   x, times 2 to the nth power.
Xfrexp(x,pn)  splits x into a fraction between 0.5 and 0.999..., which
X         is the returned value, and a power of 2, which is stored
X         in the int pointed to by pn.  If x is 0, both parts of
X         the result are 0.
Xmodf(x,py)   splits x into integral and fractional parts, both with
X         the same sign as x.  The fraction is returned and the
X         integral part is stored in the float pointed to by py.
Xfmod(x,y)    remainder of x/y, with the same sign as y.
Xatof(ps) converts the string ps into a float.
X
XThe file MATH.H also defines several mathematical constants.
SHAR_EOF
chmod 0600 SPECS || echo "restore of SPECS fails"
sed 's/^X//' << 'SHAR_EOF' > SQRT.C &&
X/* SQRT function for Sozobon C.                          */
X/* Copyright = David Brooks, 1989 All Rights Reserved
         */
X/*                                        */
X/* The method is from Hart et al: "Computer Approximations".  We reduce
     */
X/* the range to [0.25..1.0], then apply the polynomial:
             */
X/* 0.25927688 + 1.0520212n - 0.31632214n^2, giving 2.3 digits
         */
X/* approximation.  Then we do two Newton's iterations (giving ~9 digits
     */
```

```
X/* precision).                                          */
X/*                                        */
X/* A negative argument gives a return value of 0.0 and sets EDOM.
     */
X
X#include <fplib.h>
X#include <errno.h>
X
X/* Sadly we can't use register for objects of type fstruct. */
X
Xfloat sqrt(n)
Xfstruct n;
X{   register int exp;       /* Separate unbiased exponent.
     */
X    fstruct s;
X
X    if (n.sc[3] == 0)       /* Check for 0.0       */
X         return 0.0;
X
X    if ((exp = n.sc[3]) < 0)    /* Test argument's sign.
     */
X    {    errno = EDOM;
X         return 0.0;
X    }
X
X    exp -= BIAS;            /* Get unbiased exponent.   */
X    if (exp & 1)
X    {    n.sc[3] = BIAS-1;  /* Convert to 0.25..0.5.    */
X         ++exp;
X    }
X    else
X         n.sc[3] = BIAS;         /* Or convert to 0.5..1.
     */
X
X    s.f = 0.25927688 + n.f * (1.0520212 + n.f * -0.31632214);
X
X    s.f = s.f + n.f / s.f;      /* One Newton.
     */
X    s.sc[3] -= 1;           /* (here's the divide by 2).
     */
X    s.f = s.f + n.f / s.f;      /* The other Newton.
     */
X    s.sc[3] += ((unsigned int)exp >> 1) - 1;
X                        /* Divide by 2 and insert
X                           half the original exponent.   */
X    return s.f;
X}
SHAR_EOF
chmod 0600 SQRT.C || echo "restore of SQRT.C fails"
sed 's/^X//' << 'SHAR_EOF' > TRIG.C &&
X/* SIN, COS, and TAN functions for Sozobon C.
     */
X/* Copyright = David Brooks, 1989 All Rights Reserved
```

```
         */
X
X#include <fplib.h>
X#include <errno.h>
X
X/* SIN:
X/*                                          */
X/* The method is from Hart et al: "Computer Approximations". It reduces
     */
X/* the range to 0..pi/2, scaled to 0..1, and applies:
         */
X/* 1.57079629t - 0.64596336t^3 + 0.0796884805t^5 - 0.0046722279t^7
     */
X/*       + 0.00015082056t^9 (precision of 8.5 digits) and
     */
X/* finally fixes the sign.                          */
X/*                                          */
X/* MODIFICATIONS:                               */
X/* DWB  03/15/89 Don't try to negate x if it's zero.
     */
X/* DWB  04/30/89 Fix cos(0.0)                       */
X
Xfloat sin(a) fstruct a;
X{   register float         t, ipart, tsq;
X    fstruct            x;
X    register unsigned long  tint;
X    register char      sign;
X
X    sign = a.sc[3] & 0x80;
X    a.sc[3] ^= sign;       /* sin(negative x) = -sin(-x) */
X
X/* If the argument is huge, the result is rather arbitrary -- and,
     */
X/* besides, the following code will break.
     */
X
X    if (a.sc[3] >= (BIAS + MANT_BITS))
X    {    errno = ERANGE;
X         return 0.0;
X    }
X
X    t = a.f * M_2_PI;      /* Express in quarter-turns */
X
X/* Get integer part and fraction.  Optimise the common case, in the
X   first quadrant */
X
X    if (t <= 1.0)
X         tint = 0L;
X    else
X    {    tint = t;          /* tint = whole quarterturns */
X         ipart = (float)tint;
X         t = (tint & 1)?ipart - t + 1.0:t - ipart;   /* Get fraction */
X    }
```

```
X    tsq = t * t;
X    x.f = t * (1.57079629 + tsq * (-0.64596336 + tsq * (0.0796884805
X         + tsq * (-0.46722279e-2 + tsq * 0.15082056e-3))));
X
X    if (x.sc[3] != 0)           /* Don't negate zero!  */
X    {    if ((int)tint & 2)     /* Quadrants 3 and 4   */
X              x.sc[3] |= 0x80;  /* Set negative        */
X         if (sign != 0)
X              x.sc[3] ^= 0x80;  /* Negate     */
X    }
X
X    return x.f;
X}
X
X/* COS:                                        */
X/*                                        */
X/* Although approximations are available, this will do fine.  I'm not
     */
X/* proud of the hack to fix the only rational argument with a rational
     */
X/* result, but then pride is the first deadly sin...
          */
X
Xfloat cos(a) float a;
X{
X    return a==0.0?1.0:sin(M_PI_2 - a);
X}
X
X/* TAN:                                        */
X/* (v-e-r-y s-l-o-w-l-y)                           */
X
Xfloat tan(a) float a;
X{
X    return sin(a) / sin(M_PI_2 - a);
X}
SHAR_EOF
chmod 0600 TRIG.C || echo "restore of TRIG.C fails"
rm -f s2_seq_.tmp
echo "You have unpacked the last part"
exit 0


From 4508866298569821651
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,2ad253920d4cbe32,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-06-21 22:17:37 PST
Path: utzoo!attcan!utgpu!jarvis.csri.toronto.edu!rutgers!tut.cis.ohio-state.edu!
ucbvax!agate!saturn!koreth@ssyx.ucsc.edu
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i043: decky -- Connect DEC keyboard to ST
Keywords: shar, 68000
```

Message-ID: <7950@saturn.ucsc.edu>
Date: 14 Jun 89 07:20:30 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 578
Approved: koreth@ssyx.ucsc.edu

Submitted-by: mark@ncsuvx.ncsu.edu (Mark Boyd)
Posting-number: Volume 2, Issue 43
Archive-name: decky

     Here is the sources version of the DEC keyboard stuff.
If you haven't seen my note with the binary version, please
read that before dealing with this.

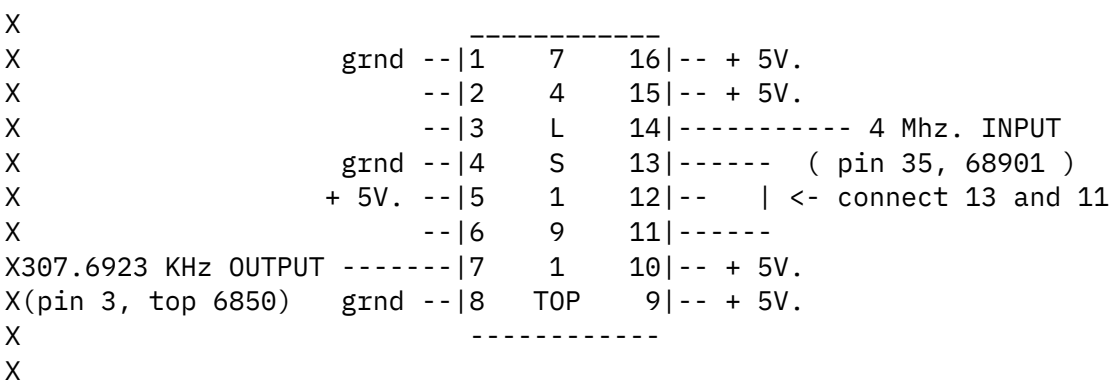
```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      DECKEY.LIS
#      MIDIKB.S
#      SERIALKB.S
#
sed 's/^X//' << 'SHAR_EOF' > DECKEY.LIS &&
XDECKeybd.doc                                                  June, 1989
X
XIntroduction:
X
XI'm typing this on a DEC LK201 keyboard connected to an Atari 1040ST
Xthrough the midi port.  Now I won't have to get a Mega to have a
Xdecent, detachable, keyboard!  Besides, this keyboard has a better
Xlayout and ( IMHO ) feel than the one that comes with a Mega.
X
XOne of reason that I started using ST's was that its keyboard layout is
Xalmost identical to a DEC keyboard.  Since my microcomputers spend a lot
Xof time acting as terminals talking to VAX computer systems, the
Xcompatible keyboard layout is very important to me.  On the other hand,
XI have always hated the feel of  the ST keyboard.  I think it is the
XST's worst feature.  Now I have been able to keep the layout and get
Xmuch better feel in a detached keyboard.
X
XLK201s keyboards are used on all current DEC products, including
Xterminals and workstations. Moreover, they were used on the DEC Rainbow
XPC's and on the previous generation of DEC terminals.  As a result,
Xthey are fairly common at ham and computer fests. You could also get a
Xnew one from DEC as a replacement part, albeit for too much money!
X
XThe obvious choice for an external ST keyboard would be an extended PC
Xkeyboard.  Its layout is fairly close to the ST's, and it is relative
Xcheap and very available.  Unfortuately, its electrical interface is
Xquite incompatible with any of the Atari's ports. PC keyboard use with
Xthe ST would require a complicated external interface adapter.  I don't
Xknow if anyone has done it, but I decided it wasn't worth the trouble.
XOn the other hand, the LK201 keyboard can be wired directly to either
```

Xthe serial or midi ports.
X
XLK201s conform to RS-423 standards at 4800 baud, 8 bits, no parity, one
Xstop bit. They send a single byte of code (a scan code) for most key
Xpresses, only the shift and control keys send codes on both down and
Xup. When a key or keys is help down, a metronome (repeat) code is
Xsent at a fixed rate.
X
XThe ST keyboards use a serial interface at 7.8125 K baud. They send a
Xscan code when a key is depressed or released. The software require to
Xconvert input from the DEC keyboard to that used by the ST is straight
Xforward.
X
XIf you are willing to use your serial port, no modifications to the ST
Xor keyboard are required. All it takes is a small, external, power
Xsupply and a RJ11 (modular phone plug) to DB25 adapter. The
Xlimitation when using the serial port is that you can't use the ST as a
Xterminal with the external keyboard. The midi port interface doesn't
Xhave this limitation but it does require internal modifications to the
XST.
X
XThe power supply requirement is 200ma. at 12 V. The voltage is
Xregulated internally so it isn't very critical. A range from 10 V. to
X15 V. seems OK, although you might have internal heat problems in the
Xkeyboard at the upper end of that range. Suitable wall transformer type
Xsupplies are available for less than $10. Since I'm not going to buy a
XMega ST, my ST's power supply will have to handle 4 megabytes of memory
Xin the near future. I didn't want to further load the ST's already
Xmarginal power supply with the extra keyboard.
X
XThe midi port problem is that its baud rate is is hardwired into the
Xcomputer. Changing the midi port's baud rate to 4800 baud is
Xstraightforward, but it should only be done by someone with good
Xsoldering skills and PC (that's Printed Circuit, not Personal
XComputer!) experience.
X
XThe modification uses one TTL IC, half a dozen short pieces of wire,
Xtwo solder connections to pins on existing IC's and one trace cut on
Xthe top of the ST circuit board. It isn't even necessary to remove the
Xmetal shield on a 1040 ST, because everything you need to get at is
Xlocated under the disk drive. Since this change will affect the
Xoperation of the midi port, you may wish to mount a switch (SPDT)
Xthat selects either 4800 or the normal (31.25K)baud rate.
X
XNo external keyboard, unless it is simply an ST keyboard mounted in a
Xseparate case, can replace the built in keyboard. STs need the
Xmouse, joystick, and clock functions of the original keyboard. An
Xexternal keyboard must simply work in parallel with the original
Xkeyboard.
X
XSoftware:
X
XWhen you strike a key on the ST's keyboard, a scancode is sent from it

Xto a serial port on the main board. This causes the serial port to
Xgenerate an interrupt. The interrupt is serviced by a routine that:
X
X   1. Gets the scan code sent by the keyboard
X   2. Determines wether it is dealing with
X   regular keys, shift keys or the mouse
X   3. Performs the appropriate function.
X
XAs a result of a regular key press, the scancode and an ascii code
X(looked up in a set of tables) are placed in a queue. Shift codes
X(right shift, left shift, control, caps lock, and alternate) simply
Xcause a change the contents of a flag byte. The contents of this byte
Xdetermine which of three tables is used for looking up the scan code
Xand if control or ALT are used as modifiers.
X
XAn external keyboard must duplicate, as much as is possible, these
Xregular key and shift functions and not disrupt the other (mouse,
Xclock, and joystick) keyboard functions. Since the builtin keyboard
Xfunctions involve modification of data structures (keyboard queue and
Xshift byte) in memory, the external keyboard software can simply
Xmodify the same data structures. If the keyboard software was not
Xinterrupt driven with these intermediate data stuctures, it would be
Xmuch more dificult to add an external keyboard.
X
XThe serial port, the keyboard port, and the midi port all use similar
Xinterfaces. This makes it easy to write an interrupt handler for the
Xserial or midi port that duplicate the services provide by keyboard
Xport interrupt handler. Most of the needed subroutines are already in
Xthe ST's ROMs. The code needed to add a keyboard is quite small, about
X700 bytes. I'm sending uuencoded sources and binaries to
Xcomp.{sources|binaries}. atari.st.
X
XBugs/Differences from the ST keyboard:
X
XWell this an 'alpha' release, so it probably has a few unintended
X'features'! On the other hand, it is a small, simple, and straight
Xforward program. I have been using the midi version for several weeks
Xon a 1040 ST and a friend has been using the serial port version on a
X520 ST for two weeks. We haven't found any problems.
X
XI did have trouble when I tried using the external keyboard to replace
Xmy mouse buttons. I had to eliminate that 'feature' since it regularly
Xcrashed the ST! The serial version won't work with a screen blanker
Xthat monitors the keyboard/midi interrupt vector since it doesn't use
Xthat vector! The midi version works with idle 1.2, but it must be run
Xbefore idle is run.
X
XI didn't put a lot of effort into handling the alt key because I very
Xrarely use it. On the LK201, the alt key says 'Compose Character' and,
Xbecause of the way I handled it, it only affects the next key pressed.
XThe only program I use that has a problem with alt is Uniterm.
XUnfortunately I use Uniterm more than any other program (thank you
XSimon Poole!). Uniterm doesn't use the normal alt flag, so I just do

Xthe Uniterm alt stuff from my regular keyboard.
X
XThe LK201 keyboard is programmable, but I have not made any use of this
Xfeature. The midi interface is only 'RS423 compatible' for input, so
Xextra hardware would be required there. The serial interface should be
Xable to talk to the keyboard, but I had no luck in my limited attempts
Xto do so. Since I am comfortable with the keyboards default settings,
XI made my interface input only.
X
XNow let's consider the keyboard remappings: There is only one shift
Xcode, so there is no way to set only one of two shift bits. I don't
Xhave any programs that distinguish between the the two shift keys,
Xbut you may. It can't be done with the DEC keyboard. Another DEC
Xlimitation, DEC keyboards don't have a backspace key or an escape
Xkey in the main keypad. I 'fixed' both these problems.
X
XDelete is the key where backspace was and shift-delete acts as
Xbackspace. This works well when I use the ST as a terminal and with
Xemacs like editors. Since some ST programs use backspace as delete
Xleft, I also put backspace and delete on the two extra keys in the
Xmiddle of the pad above the cursor keys.
X
XDEC has this wierd < > key down between the z and the left shift, I
Xmoved ~` down there, and put esc up in the top left hand corner where
X~` was. Note that < and > are shift-, and shift-. respectively.
XActually DEC supports this change in their new 300 series of terminals.
XThey will even sell you a couple of new keytops to make it look
Xpretty. I just marked < and > on the ',' and '.' keycaps.
X
XA few other keys have moved around a bit, but that is about it. There
Xare 20 function keys, so I mapped the top ten to the shifted function
Xkey codes. The keypad is identical, except for the notation on the top
Xfour keys (Pf1 - Pf4 instead of ( ) / *) and the kp+ key that says
X',' . There really is very little difference, and I find no problem,
Xother that feel, in switching between the two keyboards. After typing
Xon the DEC keyboard, the ST feels very bad! Now, on to the hardware
Xdetails.
X
XHardware:
X
XThe RJ11 (standard modular telephone connector) plug on the DEC
Xkeyboard has four wires. They are color coded black, red, green, and
Xyellow. The red wire supplies power (+12 V.) to the keyboard. It
Xshould be connected to the positive side of the power supply. The
Xgreen wire is ground for both the power supply and the communications
Xlines. It must be connnected to the negative side of the power supply.
XThe black wire is the RS-423 output from the keyboard. The yellow wire
Xis the RS-423 input to the keyboard. This interface does not use the
Xyellow wire.
X
XFor the serial port, I used an RJ11 to DB25 adapter that looks like an
Xoversized DB25 shell. Our computer center uses them for all our
Xterminals so that they can use RJ11 wall jacks. I brought the power

Xwires in through a small hole that I melted in the shell, and did all
Xthe wiring inside. There were originally four wires connected to pins
X2, 3, 7, and 20. I cut all the wires and reconnected the ones for 3 (
Xblack wire from keyboard) and 7 (green wire from keyboard and power
Xsupply negative lead). Plug the keyboard into the RJ11 jack.
X
XPlug the DB25 into the serial port, power the keyboard, and run
Xserialkb.prg. Serialkb.prg installs itself as the serial interrupt
Xdriver the first time it is run. It deinstalls itself when it is run
Xagain and restores the original serial port interrupt handler. The
Xsmall amount of memory used (< 1KB) will not be released until you
Xreboot. It will create a new version of itself every other time you
Xrun it.
X
XThe midi port (IN) connector is a five pin DIN male. I actually used
Xthe end of a cable from a defuct IBM PC keyboard. Connect the black
Xwire from the keyboard to pin 4 and the green wire to pin 5. Watch out
Xfor the wierd ordering, it isn't 1 2 3 4 5! The keyboard power
Xconnections are the same as for the serial port (positive voltage to
Xred wire, negative to green). Note that the midi interface is not
XRS-423 compatible, but that a RS-423 output should drive a midi input.
XIt works well on my system, but 'your mileage may vary'. Now lets go
Xthrough the hard part, changing the midi port's baud rate to 4800 baud.
X
XBefore starting this modification, reflect on the fact you will be
Xviolating any Atari warantee and that you could, with suitable
Xcarelessness or ignorance, trash your ST. Don't try this, or any
Xother hardware modification, unless you know what you are doing and
Xare able to handle any damage you may cause. I obviously can't help
Xyou if you damage your system. I hereby disclaim any and all
Xresponsiblility for the correctness of the following instructions.
XYou should assume that they are designed to destroy your system 8^)!
XThese instructions are not a cookbook, they assume that you know
Xwhat you are doing. If you don't, find somebody who does and get
Xthem to help you!
X
XThe midi port uses an ACIA. In its stock form the clock input to the
XACIA is 500 KHz and the ACIA is programmed to divide that clock by 16
Xgiving a 31.25 K baud rate. The keyboard ACIA uses the same clock, but
Xit is programmed to divide it by 64, giving 7.185 K baud. To get 4800
Xbaud we need 16 or 64 times 4800 Hz at the midi ACIA clock input. The
Xsystem has 8, 4, 2, and .5 Mhz clock signal available.
X
XA note of credit here, most of this information and the ROM listings
Xneeded to complete this project came from 'Atari ST Internals',
Xpublished by Abacus. This is, by far, the most useful ST book I own.
X
XA little playing around with a calculator shows 4MHz, divided by 13,
Xthen divided by 64, gives 4807.7 Hz. Good enough! There is a 4 MHz
Xclock input to the 68901 MFP chip which is located near the ACIAs. Now
Xall we need is a ciruit to divide that 4 Mhz clock by thirteen.
X
XDividing a TTL level clock signal by 13 is easy to do with a 4 bit

Xbinary TTL counter chip with a parallel load function. After a little
Xreading in my well worn 'TTL Cookbook' by Don Lancaster, and a few
Xminutes spent sorting through the chips I had available, I came up with
Xthe following circuit:
X
X

grnd --|1 7 16|-- + 5V.
--|2 4 15|-- + 5V.
--|3 L 14|---------- 4 Mhz. INPUT
grnd --|4 S 13|------ (pin 35, 68901)
+ 5V. --|5 1 12|-- | <- connect 13 and 11
--|6 9 11|------
307.6923 KHz OUTPUT -------|7 1 10|-- + 5V.
(pin 3, top 6850) grnd --|8 TOP 9|-- + 5V.
X
XI wired this circuit directly on the pins of the 74LS191 IC using
Xwire-wrap wire. I soldered it after checking to make sure that it
Xdid indeed divide by 13. At first it didn't because I had wired pin
X11 to pin 12, intead of to pin 13! I always check and double check
Xevery step of my work. Careful checking along the way is tedious,
Xbut it prevents much pain and suffering later.
X
X
XNow, for the work inside the ST. Take it apart and remove the floppy
Xdrive (this is for a 1040, I think the relevant chips are located in
Xthe same area on the 520, near the serial port connector, but I
Xhaven't made this modification to a 520. Locate the MFP and the two
XACIAs. On my 1040, the upper (nearest the back) ACIA is used for
Xthe midi port. There is a small trace connecting pins 3 and 4 (the
Xclock inputs) of both ACIA chips. Cut it somewhere between the two
XACIAs. Be careful to completely sever that trace and not to damage
Xany other traces.
X
XTo mount the 74LS191, bend pins 1 and 16 up above the top of the chip
Xand solder them to the leads of a bypass capacitor located above and
Xbetween the ACIA and MFP chips. This holds the chip in place (upside
Xdown) and provides power and ground connections. It is easy to break
Xthe pins, so be gentle. Then carefully connect the INPUT by soldering
Xwire wrap wire directly to pin 35 of the MFP. This pin is located
Ximmediately below the Vss (ground) pin. The ground is easy to
Xidentify because it is a thick trace. Finally, connect the OUTPUT to
Xpin 3 of the midi ACIA using wire wrap wire.
X
XCheck and double check the wiring and the mounting of the TTL IC. Make
Xsure that none of its leads can touch anything conductive, for example,
Xthe bottom shield on the disk drive! Then put it all back together,
Xand boot it. Be prepared to go back inside and fix any errors!
X
XConnect the keyboard to the midi port, power the keyboard, run the
Xprogram midikb.prg, and type on your new keyboard! Midikb installs
Xitself as the midi port interrupt handler. It uses 1 K of memory.
XSince it does not interfere with the normal use of the system (as

```
Xfar as I have been able to determine 8*)), I always have it
Xinstalled, but only power up the keyboard when I am using it.  To
Xdeinstall, you must reboot the system.
X
X
X    Mark Boyd, Computer Science
X    Univ. of North Carolina at Asheville
X    Asheville, NC 28804
X    mbunca@ecsvax.uncecs.edu
X    mark@ncsuvx.ncsu.edu
X
SHAR_EOF
chmod 0600 DECKEY.LIS || echo "restore of DECKEY.LIS fails"
sed 's/^X//' << 'SHAR_EOF' > MIDIKB.S &&
Xstart:
X         pea      install(pc)      ; superexec to set int vector
X         move.w   #38,-(a7)        ; and initialize midi ACIA
X         trap     #14
X         addq.l   #6,a7
X         clr.w    -(a7)
X         move.l   #$400,-(a7)      ; reserve space, keep process
X         move.w   #$31,-(a7)
X         trap     #1               ; done
Xinstall:
X         pea      handler(pc)
X         move.l   (a7)+,$118       ; pointer keyboard/midi intrpt
X         move.b   #3,$fffffc04     ; master reset midi acia
X         move.b   #$96,$fffffc04   ; set for /64, 8bit, 1 s, no parity
X         rts
Xhandler:
X         movem.l  d0-d7/a0-a6,-(sp); save registers
X         lea      $0,a5            ; clear a5 ( base page pointer )
Xmore:    jsr      midisrv
X         move.l   $dec(a5),a2      ; get keyboard addr
X         jsr      (a2)             ; service keyboard
X         btst     #4,$fffffa01     ; still interrupt
X         beq.s    more             ; yes
X         bclr     #6,$fffffa11     ; clear int service bit
X         movem.l  (sp)+,d0-d7/a0-a6
X         rte
Xmidisrv: move.b   d0,-(sp)
X         move.l   $de8(a5),a2      ; get midi addr
X         jsr      (a2)             ; service midi
X         move.b   (sp)+,d3         ; see if we got something
X         cmp.b    d0,d3            ; kludge to check for input
X         bne.s    doit
X         rts
Xdoit:    andi.l   #$ff,d0
X         move.l   d0,d3
X         cmp.b    #$55,d0          ; ignore codes below $56
X         bls      done
X         cmp.b    #$8d,d0          ; prevent alt insert
X         bne.s    noinsr           ; to avoid crashing!
```

```
X         btst    #3,$e1b          ; is it alt?
X         beq     lookup           ; nope
X         bra     done             ; yes, toss it
Xnoinsr:  cmp.b   #$b4,d0          ; is it metronome?
X         bne.s   norept
X         move.l  oldkey,d0        ; repeat last key
X         bra     almost
Xnorept:  cmp.b   #$bf,d3          ; is it tilde - modify keyboard
X         bne.s   notilde
X         move.w  #$01,d0          ; make it esc
X         bra     almost
Xnotilde: cmp.b   #$c9,d3          ; is it <>
X         bne.s   nogtlt
X         move.w  #$29,d0          ; make it tilde
X         bra     almost
Xnogtlt:  cmp.b   #$bc,d3          ; is it backsp
X         bne.s   notdel
X         btst    #0,$e1b          ; is it shifted
X         bne.s   notdel
X         move.w  #$53,d0          ; make it del
X         bra     almost
Xnotdel:  cmp.b   #$ad,d3          ; handle shift codes
X         bmi     lookup
X         cmp.b   #$bb,d3
X         bpl     lookup           ; nope
X         move.b  $e1b,d2          ; get shift mask
X         cmp.b   #$af,d3          ; is it shift
X         bpl.s   arnd1
X         eori.b  #3,d2            ; toggle shift bits
X         bra.s   arnd5
Xarnd1:   cmp.b   #$b0,d3
X         bpl.s   arnd2
X         bchg    #2,d2            ; toggle ctrl bit
X         bra.s   arnd5
Xarnd2:   cmp.b   #$b1,d3
X         bpl.s   arnd3
X         bchg    #4,d2            ; toggle caps lock bit
X         bra.s   arnd5
Xarnd3:   cmp.b   #$b2,d3
X         bpl.s   arnd4
X         bset    #3,d2            ; set alt bit
X         bra.s   arnd5
Xarnd4:   cmp.b   #$b4,d3          ; all ups
X         bpl.s   arnd5
X         andi.b  #$f0,d2          ; clear shift,ctrl, and alt bits
Xarnd5:   move.b  d2,$e1b          ; save new shift status
X         bra.s   done
Xlookup:  lea     table(pc),a0     ; convert dec to st scan code
X         sub.w   #$56,d3
X         and.l   #$ff,d3
X         asl.l   #1,d3
X         adda.l  d3,a0
X         move.w  (a0),d0
```

```
Xalmost:  move.l   #$db0,a0         ; point to kbd iorec addr
X         lea      $0,a5
X         move.l   d0,oldkey
X         jsr      $fc2aa6          ; get scan code in buffer
X         bclr     #3,$e1b          ; clear alt shift
Xdone:    rts
Xtable:   dc.w     $3b,$3c,$3d,$3e,$3f,0,0,0,0,0,0,0,0,0 ; function keys
X         dc.w     $40,$41,$42,$43,$44,0,0,0,0,0,0,0,0
X         dc.w     $54,$55,$56,$57,0,0,0,0,0,0,0   ; shifted funct keys
X         dc.w     $58,$59,0,0,$5a,$5b,$5c,$5d,0,0,0,0,0,0
X         dc.w     $62,$0e,$61,$52,$0e,$47,0,0     ; keys abv cursor pad
X         dc.w     $70,0,$71,$72,$6d,$6e,$6f,$6a   ; numeric keypad
X         dc.w     $6b,$6c,$4e,$67,$68,$69,$4a
X         dc.w     $63,$64,$65,$66,0,0             ; pf keys
X         dc.w     $4b,$4d,$50,$48                 ; cursor keys
X         dc.w     0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
X         dc.w     $0e,$1c,$0f,$29,$02,$10,$1e,$2c,0 ; main keyboard
X         dc.w     $03,$11,$1f,$2d,0,0,$04,$12,$20,$2e
X         dc.w     0,$05,$13,$21,$2f,$39,0,$06,$14,$22,$30
X         dc.w     0,$07,$15,$23,$31,0,$08,$16,$24,$32,0
X         dc.w     $09,$17,$25,$33,0,$0a,$18,$26,$34,0
X         dc.w     $0b,$19,0,$27,$35,0,$0d,$1b,$2b,0
X         dc.w     $0c,$1a,$28,0,0,0
Xoldkey:  dc.l     0
X         end
X
SHAR_EOF
chmod 0600 MIDIKB.S || echo "restore of MIDIKB.S fails"
sed 's/^X//' << 'SHAR_EOF' > SERIALKB.S &&
Xstart:
X         pea      install(pc)      ; superexec to set int vector
X         move.w   #38,-(a7)        ; and check for prior installation
X         trap     #14
X         addq.l   #6,a7
X         cmp.b    #1,d0
X         bne.s    bye
X         move.w   #-1,-(sp)        ; setup rs232 port
X         move.w   #-1,-(sp)
X         move.w   #-1,-(sp)
X         move.w   #-1,-(sp)
X         move.w   #0,-(sp)         ; no handshake
X         move.w   #2,-(sp)         ; 4800 baud
X         move.w   #15,-(sp)
X         trap     #14
X         add.l    #14,sp
Xbye:      clr.w    -(a7)
X         move.l   #$400,-(a7)     ; reserve space, keep process
X         move.w   #$31,-(a7)
X         trap     #1
Xinstall:
X         move.l   $130,a0         ; get the serial port int vector
X         movea.l  -4(a0),a1       ; check to see if serialkb installed
X         cmp.l    #$6b796264,a1
```

```
X         bne.s    hookit          ; no, install it
X         move.l   -8(a0),$130     ; yes, restore original vector
X         move.b   #0,d0
X         rts
Xhookit:  move.l   a0,oldvec       ; save the original vector
X         pea      handler(pc)     ; and set the new vector
X         move.l   (a7)+,$130      ; pointer to rs232 in buf full intrpt
X         move.b   #1,d0
X         rts
Xoldvec:  dc.l     0
Xmagic:   dc.b     'kybd'
Xhandler:
X         movem.l  d0-d3/a0-a5,-(sp); save registers
X         lea      $fffffa01,a1     ; pointer to mfp
X         btst     #7,42(a1)        ; is it receive int
X         beq      done
X         clr.l    d0
X         move.b   46(a1),d0        ; get data
X         move.l   d0,d3
X         cmp.b    #$55,d0          ; ignore codes below $56
X         bls      done
X         cmp.b    #$8d,d0          ; prevent alt insert
X         bne.s    noinsr           ; to avoid crashing!
X         btst     #3,$e1b          ; is it alt?
X         beq      lookup           ; nope
X         bra      done             ; yes, toss it
Xnoinsr:  cmp.b    #$b4,d0          ; is it metronome?
X         bne.s    norept
X         move.l   oldkey,d0        ; repeat last key
X         bra      almost
Xnorept:  cmp.b    #$bf,d3          ; is it tilde - modify keyboard
X         bne.s    notilde
X         move.w   #$01,d0          ; make it esc
X         bra      almost
Xnotilde: cmp.b    #$c9,d3          ; is it <>
X         bne.s    nogtlt
X         move.w   #$29,d0          ; make it tilde
X         bra      almost
Xnogtlt:  cmp.b    #$bc,d3          ; is it backsp
X         bne.s    notdel
X         btst     #0,$e1b          ; is it shifted
X         bne.s    notdel
X         move.w   #$53,d0          ; make it del
X         bra      almost
Xnotdel:  cmp.b    #$ad,d3          ; handle shift codes
X         bmi      lookup
X         cmp.b    #$bb,d3
X         bpl      lookup           ; nope
X         move.b   $e1b,d2          ; get shift mask
X         cmp.b    #$af,d3          ; is it shift
X         bpl.s    arnd1
X         eori.b   #3,d2            ; toggle shift bits
X         bra.s    arnd5
```

```
Xarnd1:   cmp.b    #$b0,d3
X         bpl.s    arnd2
X         bchg     #2,d2            ; toggle ctrl bit
X         bra.s    arnd5
Xarnd2:   cmp.b    #$b1,d3
X         bpl.s    arnd3
X         bchg     #4,d2            ; toggle caps lock bit
X         bra.s    arnd5
Xarnd3:   cmp.b    #$b2,d3
X         bpl.s    arnd4
X         bset     #3,d2            ; set alt bit
X         bra.s    arnd5
Xarnd4:   cmp.b    #$b4,d3          ; all ups
X         bpl.s    arnd5
X         andi.b   #$f0,d2          ; clear shift,ctrl, and alt bits
Xarnd5:   move.b   d2,$e1b          ; save new shift status
X         bra.s    done
Xlookup:  lea      table(pc),a0     ; convert dec to st scan code
X         sub.w    #$56,d3
X         and.l    #$ff,d3
X         asl.l    #1,d3
X         adda.l   d3,a0
X         move.w   (a0),d0
Xalmost:  move.l   #$db0,a0         ; point to kbd iorec addr
X         lea      $0,a5
X         move.l   d0,oldkey
X         jsr      $fc2aa6          ; get scan code in buffer
X         bclr     #3,$e1b          ; clear alt shift
Xdone:    bclr     #4,14(a1)        ; clear int service bit
X         movem.l  (sp)+,d0-d3/a0-a5; restore registers
X         rte
Xtable:   dc.w     $3b,$3c,$3d,$3e,$3f,0,0,0,0,0,0,0,0,0 ; function keys
X         dc.w     $40,$41,$42,$43,$44,0,0,0,0,0,0,0,0
X         dc.w     $54,$55,$56,$57,0,0,0,0,0,0,0   ; shifted funct keys
X         dc.w     $58,$59,0,0,$5a,$5b,$5c,$5d,0,0,0,0,0,0
X         dc.w     $62,$0e,$61,$52,$53,$47,0,0     ; keys abv cursor pad
X         dc.w     $70,0,$71,$72,$6d,$6e,$6f,$6a   ; numeric keypad
X         dc.w     $6b,$6c,$4e,$67,$68,$69,$4a
X         dc.w     $63,$64,$65,$66,0,0              ; pf keys
X         dc.w     $4b,$4d,$50,$48                  ; cursor keys
X         dc.w     0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
X         dc.w     $0e,$1c,$0f,$29,$02,$10,$1e,$2c,0 ; main keyboard
X         dc.w     $03,$11,$1f,$2d,0,0,$04,$12,$20,$2e
X         dc.w     0,$05,$13,$21,$2f,$39,0,$06,$14,$22,$30
X         dc.w     0,$07,$15,$23,$31,0,$08,$16,$24,$32,0
X         dc.w     $09,$17,$25,$33,0,$0a,$18,$26,$34,0
X         dc.w     $0b,$19,0,$27,$35,0,$0d,$1b,$2b,0
X         dc.w     $0c,$1a,$28,0,0,0
Xoldkey:  dc.l     0
X         end
X
SHAR_EOF
chmod 0600 SERIALKB.S || echo "restore of SERIALKB.S fails"
```

```
exit 0


From -2886215261034548543
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,68a69479d26d2682,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-06-25 09:10:29 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!rutgers!apple!bionet!agate!saturn!
koreth@ssyx.ucsc.edu
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i044: djreset -- Reset DeskJet Plus
Keywords: shar, C
Message-ID: <8126@saturn.ucsc.edu>
Date: 25 Jun 89 16:10:29 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 117
Approved: koreth@ssyx.ucsc.edu

Submitted-by: ucsd.edu!cacilj!close (Diane Barlow Close)
Posting-number: Volume 2, Issue 44
Archive-name: djreset


When you turn on your DeskJet Plus printer *after* your ST has been booted,
the printer and the ST won't talk to each other and you can't print
anything.  The only way, until now, of solving this problem was to reboot
the ST (leaving the printer on).

When the printer has been turned on *before* the ST is booted, they will
talk to each other just fine.  As long as you don't turn off (and on again)
your printer, you'll be just fine.  I, for one, don't like to leave my
printer on all the time.  I like to be able to ``compute away'' and turn on
the printer only when I actually need to print something.  So far, with a
DeskJet Plus, this has been impossible.

My husband (Paul Close) and I wrote the following little desk accessory to
``reset'' the printer port and get the DeskJet and the ST talking again.
(That's not exactly what it does, but it's close enough.)  Much thanks to
Moshe Braner for his desk accessory template.

We've included the source code so that you can modify and ``pretty-up'' the
DA to your liking.  Currently it just blinks the screen when a reset is
performed--we didn't add any desktop messages saying ``printer reset'' or
anything like that.

#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      DJ_RESET.C
#
```

```
sed 's/^X//' << 'SHAR_EOF' > DJ_RESET.C &&
X/*  Reset your DeskJet Plus printer from the desktop.
X    Desk accessory written by Diane & Paul Close.
X*/
X
X#include <osbind.h>
X#include <obdefs.h>
X#include <gemdefs.h>
X
X#define TRUE  1
X#define FALSE 0
X
Xextern int gl_apid;
X
Xint menu_id, event, ret;
Xint xdesk, ydesk, hdesk, wdesk;
Xint msgbuff[8];
Xint BUSY;
Xint STATUS7;
Xint STATUS14;
X
Xchar    menusave[1520];
X
Xmain()
X{
X    int  i;
X    char *cp;
X    /* initialize printer reset options */
X     int super();
X    Cconws ("\033f");  /* disable vt52 cursor */
X    appl_init();
X    menu_id = menu_register (gl_apid,"  DJ Reset");
X    wind_get (0, WF_WORKXYWH, &xdesk, &ydesk, &wdesk, &hdesk);
Xdaloop:
X    event = evnt_multi (MU_MESAG,
X              0,0,0,0,0,0,0,0,0,0,0,0,0,
X              msgbuff,
X              0,0, &ret,&ret,&ret,&ret,&ret,&ret);
X    /* could have simply used evnt_mesag instead:
X         evnt_mesag (msgbuf);        */
X    if ((event & MU_MESAG) == 0
X         || msgbuff[0] != AC_OPEN
X         || msgbuff[4] != menu_id )
X              goto daloop;
X    graf_mouse(M_OFF,0x0L);
X    cp = (char *) Physbase();
X    for (i=0; i<1520; i++)
X         menusave[i] = *cp++;
X    wind_update(TRUE);
X    form_dial (FMD_START,
X              xdesk, ydesk, wdesk, hdesk,
X              xdesk, ydesk, wdesk, hdesk);
X    /* enable vt52 cursor if needed */
```

```
X    /* reset dj here (call subroutine) */
X    Supexec(super);
X
X    /* disable vt52 cursor if on */
X    form_dial (FMD_FINISH,
X              xdesk, ydesk, wdesk, hdesk,
X              xdesk, ydesk, wdesk, hdesk);
X    wind_update (FALSE);
X    cp = (char *) Physbase();
X    for (i=0; i<1520; i++)
X         *cp++ = menusave[i];
X    graf_mouse(M_ON,0x0L);
X    goto daloop;
X}
X
Xsuper() {
X    register unsigned char *busy;
X    busy=(char *)0xffffa01L;
X    BUSY=*busy;
X    busy=(char *)0xffff8800L;
X    *busy=7;
X    STATUS7=*busy;
X    *busy=14;
X    STATUS14=*busy;
X    STATUS14 |= 0x20;
X    *busy=14;
X    *(busy+2) = STATUS14;
X}
SHAR_EOF
chmod 0600 DJ_RESET.C || echo "restore of DJ_RESET.C fails"
exit 0


From 2641168353024002465
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,28e616451162d961,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-06-25 09:15:06 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!rutgers!apple!bionet!agate!saturn!
koreth@ssyx.ucsc.edu
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i045: modedit -- Hardwire an editor into German Modula-2
Keywords: shar, C
Message-ID: <8127@saturn.ucsc.edu>
Date: 25 Jun 89 16:15:06 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 80
Approved: koreth@ssyx.ucsc.edu

Submitted-by: piet@cs.ruu.nl (Piet van Oostrum)
Posting-number: Volume 2, Issue 45
Archive-name: modedit
```

This program can be used to glue your favorite editor into the German Modula-2 system. The program starts the editor named in the environment variable EDITOR, with the cursor enabled. The environment of the parent is passed on as the modula system forgets about it. If more than 1 parameter is given, the first one is the error file, and that file is updated to get the line/char numbers in normal ASCII code.

All this wouldn't be necessary if the Modula2 compiler did it right.

I compiled this with sozobon. [Note: the backslashes in the filename near the top may have to be doubled (like \\) or C may treat them as escapes for the next character in the filename. Dunno what Sozobon does about that. -sg]

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      modify.c
#
sed 's/^X//' << 'SHAR_EOF' > modify.c &&
X#include <basepage.h>
X#include <stdio.h>
X#include <osbind.h>
X#include <string.h>
X#define DEF_EDIT "c:\bin\xmg.ttp"
Xextern char *getenv();
Xextern char *_envp;
Xmain(argc, argv)
Xint argc; char **argv;
X{
X    char *program, buf[80], command[128];
X    int status;
X    BASEPAGE *bp = _base;
X    printf ("\033e");       /* cursor on */
X    /* search for environment */
X    /* modula-2 does not pass the environment, */
X    /* so we also look in parents' environment */
X    while (bp) {
X         _envp = bp->p_env;
X         bp = bp->p_parent;
X         program = getenv("EDITOR");
X         if (program && *program) break;
X    }
X    if (!program || !*program)
X         program = DEF_EDIT;
X    command[1] = 0;
X    if (argc > 2 ) {
X         register int c;
X         char *tempname = strdup (argv[1]);
X         FILE *errfile = fopen (argv[1], "r"), *copfile;
X         strcpy (tempname + strlen(tempname) -3, "tmp");
X         copfile = fopen (tempname, "w");
```

```
X         while ((c = getc (errfile)) != EOF)
X              putc ((c>=0x10 && c<=0x19) ? c+0x20 : c, copfile);
X         fclose (errfile);
X         fclose (copfile);
X         remove (argv[1]);
X         rename (tempname, argv[1]);
X    }
X    while (--argc > 0) {
X         strcat (&command[1], (++argv)[0]);
X         strcat (&command[1], " ");
X    }
X    command[0] = strlen(&command[1]);
X    status = (int) Pexec (0, program, command, _envp);
X    if (status) {
X         fprintf (stderr, "Can't execute %s - error %d\n",
X                   program, status);
X    }
X    printf ("\033f");       /* cursor off */
X    return status;
X}
X
SHAR_EOF
chmod 0600 modify.c || echo "restore of modify.c fails"
exit 0
```

```
From -2189220500198783110
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,6fb1deafd81577a4,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-06-25 09:31:41 PST
Path: utzoo!attcan!utgpu!jarvis.csri.toronto.edu!rutgers!tut.cis.ohio-state.edu!
ucbvax!agate!saturn!koreth@ssyx.ucsc.edu
From: koreth@ssyx.ucsc.edu (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i046: overscan -- Software driver for increased screen size
Keywords: shar, C
Message-ID: <8133@saturn.ucsc.edu>
Date: 25 Jun 89 16:31:41 GMT
Sender: usenet@saturn.ucsc.edu
Lines: 362
Approved: koreth@ssyx.ucsc.edu

Submitted-by: linus.umd.edu!nixpbe!andreas (Andreas Hoepfner)
Posting-number: Volume 2, Issue 46
Archive-name: overscan

Now here is the program to initialize the BIGSCREEN from the
german magazine 'ST Magazin'. The resolution grows up to
800 * 500. It has to be startet from the 'Auto folder', just
before GEM comes up.  [This is not the same as "bigscrn,"
posted to the binaries group; this requires a hardware
modification. -sg]
```

It patches the basepage adress for the new resolution and
allocates about 69000 Bytes for it.
Also included is (wow!) the C sourcecode (for Megamax C).


```
 +-------------------------------------------------------------------------+
 |                   Andreas Hoepfner                                      |
 |                                                                         |
 |     paper mail:                              e-mail:                    |
 |                                                                         |
 | c/o/ Nixdorf Computer AG       USA:  uunet!linus!nixbur!hoepfner.pad    |
 | Abt. ET-31                     !USA: mcvax!unido!nixpbe!hoepfner.pad    |
 | Pontanusstrasse 55                                                      |
 | D-4790 Paderborn                                                        |
 | tel.: (+49) 5251 146475                                                 |
 +-------------------------------------------------------------------------+
```


```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      OVERSCAN.C
#
sed 's/^X//' << 'SHAR_EOF' > OVERSCAN.C &&
X
X/*************************************************************************
X *
X *      OVERSCAN.C      Karsten Isakovic,Berlin 28.04.89
X *
X *    MEGAMAX C Version 1.2
X */
X
X/*.......................... HIGH-RES */
X    #define HIGH_X     672
X    #define HIGH_Y     480
X    #define HIGH_OFF  1608
X/*.......................... MID-RES  */
X    #define MID_X      848
X    #define MID_Y      280
X    #define MID_OFF   5640
X/*.......................... LOW-RES  */
X    #define LOW_X      416
X    #define LOW_Y      280
X    #define LOW_OFF   5640
X
X#include <osbind.h>
Xextern char *_base;
X
Xint  inst_ok;   /* Wurde installiert oder nicht ?                       */
Xint  RezX;      /* Breite des Bildschirms                               */
Xint  RezY;      /* Hoehe   des Bildschirms                               */
Xint  BpL;       /* BytesPerLine , also Bytes pro Bildschirmzeile        */
Xlong LineA;     /* Zeiger auf LineA Variablen                           */
```

```
Xlong WorkOut;   /* Zeiger auf das WorkOut-Feld der 'v_opnwk()' aufruf-  */
X                /*   enden Funktion (hoffentlich der DESKTOP !)         */
Xlong Back;      /* RuecksprungAddresse zu dieser Funktion                */
Xlong SaveBasAdd;/* Zwischenspeicher zur Restauration des Desktops nach  */
Xlong SaveVidAdd;/* einem inkompatiblen Programm                          */
X
Xmain()
X{
Xlong  keep,prgtop;
Xint   do_setup();
X
X Supexec(do_setup);
X switch(inst_ok)
X   {
X   case 0 : Cconws("\n\rOVERSCAN Version 1.2 installiert\n\r");
X            asm { move.l  A7,prgtop(A6) }      /* A7 steht auf Ende und */
X            keep = (prgtop+200L)-(long)_base;  /* _base auf Anfang des  */
X            Ptermres( keep,0);                 /* Programms             */
X            break;
X   case 1 : Cconws("\n\r   Kein gueltiges TOS 1.4 / BETA-TOS");
X            Cconws("\n\rOVERSCAN Version 1.2 nicht installiert\n\r");
X            Cconin();
X   case 2 : Cconws("\n\rOVERSCAN Version 1.2 nicht installiert\n\r");
X            break;
X   }
X}
X
X/******************** NEGATIVE LINE-A VARIABLEN ********************/
X
X#define WKXRez          -0x2b4
X#define WKYRez          -0x2b2
X#define V_CEL_HT        -0x2e
X#define V_CEL_MX        -0x2c
X#define V_CEL_MY        -0x2a
X#define V_CEL_WR        -0x28
X#define V_REZ_HZ        -0xc
X#define V_REZ_VT        -4
X#define BYTES_LIN       -2
X#define V_PLANES        0
X#define WIDTH           2
X
X/************************ HARDWARE *********************************/
X
X#define GEMTRAP      0x088
X#define MEMTOP       0x436
X#define RESO         0x44c
X#define V_BAS_ADD    0x44e
X#define DUMPFLAG     0x4ee
X#define SYSBASE      0x4f2
X#define HARDCOPY     0x502
X#define VID_BASHIGH  0xff8201L
X#define VID_BASMID   0xff8203L
X#define VID_ISMID    0xff8207L
```

```
X
X#define ADD_LEN      0x9000L
X
Xdo_setup()
X{
X  asm   {
X        movem.l D0-A6,-(A7)
X        lea     SaveA4(PC),A0   ; GLOBAL Zeiger A4 retten
X        move.l  A4,(A0)         ;
X        move.w  #2,inst_ok(A4)  ; noch nicht installiert
X        move.b  VID_BASMID,D0   ; Testen ob der Schalter umgelegt ist
X        move    D0,D1           ; Es wird gewartet bis der VideoAddress-
X        add     #0x7d,D1        ; Zaehler bis zum Ende der normalen
XWait1:  cmp.b   VID_ISMID,D1    ; Bildschirmseite hochgezaehlt hat.
X        bne     Wait1           ; Dann wird getestet, ob er auf den
XWait2:  cmp.b   VID_ISMID,D1    ; Anfang zurueckspringt. Wenn nicht ist
X        beq     Wait2           ; der Schalter umgelegt.
X        cmp.b   VID_ISMID,D0    ;
X        beq     ende            ; Nicht umgelegt -> Beenden
X;-------------------------------------------------------------
X        move    #1,inst_ok(A4)  ; auf BETA RAMTOS testen
X        move.l  #0x170ee,A0     ;
X        cmpi.l  #0xe848cec0,(A0); LSR #4,D0 / MULU D0,D7
X        bne     RAMTOS1_4       ; Kein BETA-RAMTOS -> Teste TOS 1.4
X        bra     TOSok
XRAMTOS1_4:
X        move.l  #0x17192,A0     ;
X        cmpi.l  #0xe848cec0,(A0); LSR #4,D0 / MULU D0,D7
X        bne     ROMTOS1_4       ; Kein RAMTOS 1.4  -> Teste ROMTOS 1.4
XROMTOS1_4:
X        move.l  SYSBASE,A0      ; _sysbase
X        cmpi.w  #0x0104,2(A0)     ; os_version 1.04 ?
X        bne     ende            ; Kein ROMTOS 1.4  -> ENDE
XTOSok:
X        move    #0,inst_ok(A4)  ;
X        dc.w    0xA000          ; LineA-Init
X        move.l  A0,LineA(A4)    ; Zeiger auf Variablen retten
X        lea     OldVec(PC),A0   ; Alten GemTrapVektor retten
X        move.l  GEMTRAP,(A0)    ;
X        lea     MyGem(PC),A0    ;
X        move.l  A0,GEMTRAP      ; und neuen Vektor eintragen
X        clr.l   D0              ;
X        move.b  RESO,D0         ;
X        cmpi.w  #2,D0           ; Aufloesung HIGH ?
X        bne     ende            ; nein -> weiter
X        lea     OldHard(PC),A0  ; Alten HardcopyVektor retten
X        move.l  HARDCOPY,(A0)   ;
X        lea     MyHard(PC),A0   ;
X        move.l  A0,HARDCOPY     ; und neuen Vektor eintragen
Xende:
X        lea     Patched(PC),A0  ; PatchFlag loeschen
X        clr.l   (A0)            ;
X        movem.l (A7)+,D0-A6     ;
```

```
X        bra     end_setup       ; Ende des Setup-Teils
X;-------------------------------------------------------------------------
XSaveA4: move.l  #0,-(A7)        ; (dc.l)  Speicher fuer GLOBAL-Zeiger
XPatched:move.l  #0,-(A7)        ; (dc.l)  Speicher fuer PatchFlag
X
X        dc.b    "XBRAOVER"      ; XBRA-Protokoll  ! (Ja das ist alles !!)
XOldVec: move.l  #0,-(A7)        ; (dc.l)  Alter Vektor
XMyGem:
X        lea     Patched(PC),A0  ;
X        tst.w   (A0)            ; Ist LineA schon geaendert ?
X        bne     Normal          ;   ja -> weiter
X        cmp.w   #0x73,D0        ; Ist es ein VDI-Aufruf ?
X        bne.s   Normal          ;   nein -> weiter
X        move.l  D1,A0           ; D1 ist Addresse der VDI-Parameterfelder
X        move.l  A4,-(A7)        ; A4 Register retten
X        move.l  SaveA4(PC),A4   ; und GLOBAL-Zeiger holen
X        move.l  12(A0),WorkOut(A4);  WorkOut-Zeiger speichern
X        move.l  (A0),A0         ; VDI-Control[0] entspricht dem VDI-Befehl
X        cmpi.w   #1,(A0)        ; ist es Open-WorkStation ?
X        bne.s   Norm2           ;    nein-> weiter
X        move.l  6(A7),Back(A4)  ; Ruecksprung merken
X        lea     PatchIt(PC),A0  ; und PatchIt als Ruecksprung eintragen
X        move.l  A0,6(A7)        ;
XNorm2:  move.l  (A7)+,A4        ; A4 Register zurueckholen
XNormal:
X        move.l  OldVec(PC),A0   ;
X        jmp     (A0)            ; Orginal-Routine ausfuehren
X;-------------------------------------------------------------------------
XPatchIt:
X        move.l  SaveA4(PC),A1           ; GLOBAL-Zeiger in A1 holen !
X        ;---------------------------------------------------------------
X        move.l  #0x170ee,A0             ; auf BETA RAMTOS testen
X        cmpi.l  #0xe848cec0,(A0)        ;
X        bne     isRAMTOS1_4             ; nein -> TOS 1.4
Xis_BETATOS1_4:
X        move.l  0x522e,A0               ; BETA TOS
X        subi.l  #ADD_LEN,8(A0)          ; Memory-Free-List
X        subi.l  #ADD_LEN,MEMTOP         ;
X        move.l  #0xcec0e84f,0x170ee     ; Patch fuer Scrolling unter TOS
X                                        ; MULU D0,D7 / LSR  #4,D7  statt
X                                        ; LSR  #4,D0 / MULU Do,D7
X        bra     weiter
XisRAMTOS1_4:
X        move.l  #0x17192,A0
X        cmpi.l  #0xe848cec0,(A0)
X        bne     isROMTOS1_4
X        move.l  0x522e,A0               ; RAMTOS 1.4
X        subi.l  #ADD_LEN,8(A0)          ; Memory-Free-List
X        subi.l  #ADD_LEN,MEMTOP         ;
X        move.l  #0xcec0e84f,0x17192     ; zweimal Patchen
X        move.l  #0xcec0e84f,0x1711d6    ;
X        bra     weiter
XisROMTOS1_4:
```

```
X        move.l  0x5328,A0               ; ROMTOS 1.4
Xtst_ROMTOS:
X        tst.l   (A0)            ; letzten Eintrag der Memory-Free-List
X        beq     do_ROMTOS       ; suchen
X        move.l  (A0),A0
X        bra     tst_ROMTOS
Xdo_ROMTOS:
X        subi.l  #ADD_LEN,8(A0)          ; Memory-Free-List
X        subi.l  #ADD_LEN,MEMTOP         ;
X        ;---------------------------------------------------------------
Xweiter: clr.l   D0
X        move.b  RESO,D0         ; Jetzige Aufloesung
Xlow:    bne     mid             ; ist es LOW-RES ?
X        move.w  #LOW_X,RezX(A1) ;
X        move.w  #LOW_Y,RezY(A1) ;
X        move.w  #236,BpL(A1)    ;
X        move.l  MEMTOP,V_BAS_ADD;
X        subi.l  #0x1600L,V_BAS_ADD
X        move.l  #LOW_OFF,D0     ; Offset
X        bra     set
X        ;---------------------------------------------------
Xmid:    cmpi.w  #1,D0           ; ist es MID-RES ?
X        bne     high            ;   nein -> weiter
X        move.w  #MID_X,RezX(A1) ;
X        move.w  #MID_Y,RezY(A1) ;
X        move.w  #236,BpL(A1)    ;
X        move.l  MEMTOP,V_BAS_ADD;
X        subi.l  #0x1600L,V_BAS_ADD
X        move.l  #MID_OFF,D0     ; Offset
X        bra     set
X        ;---------------------------------------------------
Xhigh:   move.w  #HIGH_X,RezX(A1);also  HIGHRES
X        move.w  #HIGH_Y,RezY(A1);
X        move.w  #100,BpL(A1)    ;
X        move.l  MEMTOP,V_BAS_ADD;
X        addi.l  #0x2000L,V_BAS_ADD
X        move.l  #HIGH_OFF,D1    ; vor dem Bildschirmspeicher
X        asr.l   #2,D1           ; Ruecklaufstrahl auf Schwarz setzen
X        addq.l  #1,D1           ;
X        move.l  V_BAS_ADD,A0    ;
Xh_clp:  move.l  #-1,(A0)+       ;
X        dbf     D1,h_clp        ;
X        move.l  #HIGH_OFF,D0    ; Offset
X        ;---------------------------------------------------
Xset:    move.b  0x44fL,VID_BASHIGH      ; VideoAddresse setzen
X        move.b  0x450L,VID_BASMID       ;
X        move.l  V_BAS_ADD,SaveVidAdd(A1); VideoAddresse retten
X        add.l   D0,V_BAS_ADD            ; Offset anbringen
X        move.l  V_BAS_ADD,SaveBasAdd(A1); v_bas_add retten
X        move.l  #0x4400,D1              ; Bildschirmspeicher auf Schwarz
X        move.l  V_BAS_ADD,A0            ; loeschen
Xsc_clp: move.l  #-1,(A0)+               ;
X        dbf     D1,sc_clp               ;
```

```
X        bsr     setLineA(PC)            ; LineA Werte eintragen
X        move.l  WorkOut(A1),A0          ;
X        move.w  RezX(A1),0(A0)          ; Work_out[0] = Breite
X        move.w  RezY(A1),2(A0)          ; Work_out[1] = Hoehe
X        lea     Patched(PC),A0          ; den Patch nur einmal ausfuehren,
X        move.l  #-1,(A0)                ;   also setzen wir ein Flag
X        move.l  Back(A1),-(A7)          ; Zum 'v_opnwk()' - Aufrufer
X        rts                             ;   zurueckspringen
X;---------------------------------------------------------------------
XsetLineA:
X        movem.l D0/A0-A1,-(A7)
X        move.l  SaveA4(PC),A1           ; GLOBAL Zeiger in A1 holen !
X        move.l  LineA(A1),A0            ; LineA Zeiger holen
X        move.w  RezX(A1),V_REZ_HZ(A0)   ; Breite setzen
X        move.w  RezX(A1),WKXRez(A0)     ;
X        subi.w  #1,WKXRez(A0)           ; Breite-1
X        move.w  RezY(A1),V_REZ_VT(A0)   ; Hoehe   setzen
X        move.w  RezY(A1),WKYRez(A0)     ;
X        subi.w  #1,WKYRez(A0)           ; Hoehe-1
X        move.w  BpL(A1) ,BYTES_LIN(A0)  ; Bytes pro Zeile setzen
X        move.w  BpL(A1),WIDTH(A0)       ;
X        move.w  RezX(A1),D0             ;
X        asr.w   #3,D0                   ;
X        subq.w  #1,D0                   ; Breite/8 -1
X        move.w  D0,V_CEL_MX(A0)         ; -> Anzahl Buchstaben pro Zeile
X        clr.l   D0                      ;
X        move.w  RezY(A1),D0             ;
X        divu.w  V_CEL_HT(A0),D0         ;
X        subq.w  #1,D0                   ; Hoehe/Buchstabenhoehe -1
X        move.w  D0,V_CEL_MY(A0)         ; -> Anzahl Buchstabenzeilen
X        move.w  V_CEL_HT(A0),D0         ;
X        mulu.w  BpL(A1),D0              ; AnzBuchstaben*BytesProZeile
X        move.w  D0,V_CEL_WR(A0)         ; -> Blocklaenge fuer Scrolling
X        movem.l (A7)+,D0/A0-A1
X        rts
X;---------------------------------------------------------------------
X        dc.b    "XBRAOVER"              ;  HARDCOPY nur bei HIGH-RES
XOldHard:move.w  #0,-(A7)                ; (dc.l)
XMyHard: movem.l D0-D4/A0-A1/A4,-(A7)    ;
X        move.l  SaveA4(PC),A4           ; GLOBAL Zeiger holen
X        lea     SaveVidAdd(A4),A0       ; VideoAddressZeiger setzen
X        move.b  1(A0),VID_BASHIGH       ;
X        move.b  2(A0),VID_BASMID        ;
X        move.l  SaveBasAdd(A4),V_BAS_ADD; v_bas_add setzen
X        bsr     setLineA(PC)            ; LineA nochmal setzen (wird zB von
X                                        ;                DEGAS veraendert )
X        clr.l   D0                      ;
X        move.l  #-1L,D4                 ; Loeschwert (Schwarze Pixel)
X        move.l  SaveVidAdd(A4),A1       ;
X        move.l  SaveBasAdd(A4),A0       ;
X        subq.l  #2,A0                   ; Falls HIGH_OFF nicht /4 teilbar
XHflp:   move.l  D4,(A1)+                ; Vor dem Bildschirm Ruecklauf
X        cmpa.l  A0,A1                   ; Pixel auf Schwarz setzen
```

```
X        blt     Hflp                    ;
X        ;---------------------------------
X        move.l  SaveBasAdd(A4),A0       ;
X        clr.l   D0                      ; Den ungenutzten Bereich im
X        clr.l   D1                      ; Ruecklauf auf Schwarz setzen
X        move    BpL(A4),D0              ;
X        move    RezX(A4),D1             ; BytesProZeile-Breite/8
X        asr     #3,D1                   ; -> Breite des Bereichs
X        sub     D1,D0                   ;
X        move    RezY(A4),D2             ; Hoehe des Bereichs
X        subq    #1,D2                   ;
X        subq    #1,D0                   ;
XHlp1:   adda.l  D1,A0                   ; Normalen Bildbereich ueberspringen
X        move    D0,D3                   ;
XHlp2:   move.b  D4,(A0)+                ; Bereich loeschen
X        dbf     D3,Hlp2                 ;
X        dbf     D2,Hlp1                 ;
X        ;---------------------------------
X        move.l  #250,D0                 ; 1000 Bytes weiterloeschen
X        move.l  A0,D0                   ;
X        andi.l  #0xfffffeL,D0           ;
X        move.l  D0,A0                   ;
XHlp3:   move.l  D4,(A0)+                ;
X        dbf     D0,Hlp3                 ;
X        move    #-1,DUMPFLAG            ; DumpFlag loeschen
X        movem.l (A7)+,D0-D4/A0-A1/A4
X        rts
X;-------------------------------------------------------------------------
Xend_setup:
X        }
X}
SHAR_EOF
chmod 0600 OVERSCAN.C || echo "restore of OVERSCAN.C fails"
exit 0


From -6247957308378019976
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,ac46cef8ca86b313,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-19 14:42:46 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
agate!apple!sun-barr!newstop!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i047: gnutar -- GNU's version of UNIX "tar"  part01/08
Keywords: shar, C
Message-ID: <33887@grapevine.uucp>
Date: 19 Jul 89 21:42:46 GMT
Sender: news@grapevine.uucp
Lines: 1757
Approved: koreth%panarthea.ebay@sun.com
```

Submitted-by: 7103_300@uwovax.uwo.ca (Eric R. Smith)
Posting-number: Volume 2, Issue 47
Archive-name: gnutar/part01

   Here is GNU Tar 1.04 ported to TOS. It has all of the
features that are applicable on a single tasking machine;
it can read, write, and create files that are completely
compatible with the Unix tar utility (even the timestamps
are the same). This version is more recent than the one
distributed with the GCC; in particular, a minor bug in
the interpretation of the TZ environment variable is fixed,
and the "stat" function returns more sensible read/write/execute
flags (and can now be used on "." and "..").
   For those not familiar with "tar", it is
an archive utility, widely used on Unix systems. It can
be used to store whole filesystems. Unlike zoo and arc,
it does not do compression; under Unix the "compress"
utility is often applied to tar files, and the same could
easily be done on the ST. GNU tar has a number of extra
features, including the ability to create multi-volume
tar files. The files readme.st and tar.tex in the ARC file
contain more information.
   In accordance with the GNU license, the complete source code
(in C) is included, as well as the executable file gnutar.ttp.
[The binary will appear in comp.binaries.atari.st. -sg]
gnutar was compiled with the GCC v1.34, with the new improved
library (in fact, a newer more improved version :-).
--
Eric R. Smith   email:
Dept. of Mathematics   7103_300@uwovax.uwo.ca
University of Western Ontario   7103_300@uwovax.bitnet
London, Ont. Canada N6A 5B7   (a shared mailbox: put my name on
ph: (519) 661-3638   the Subj: line, please!)

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
# This is part 1 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
#
#    Run the following text with /bin/sh to create:
#      BUFFER.C
#      COPYING
#      CREATE.C
#      DIFFARCH.C
#      EXTRACT.C
#      GETDATE.Y
#      GETOLDOP.C
#      LIST.C
#      MAKEFILE
#      MSD_DIR.C
#      MSD_DIR.H
#      NAMES.C
```

```
#      OPEN3.H
#      PORT.C
#      PORT.H
#      README
#      README.ST
#      RMT.H
#      RTAPE_LI.C
#      RTAPE_SE.C
#      TAR.C
#      TAR.H
#      TAR.TEX
#      UPDATE.C
#      VERSION.C
#      WILDMAT.C
#
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
echo "x - extracting BUFFER.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > BUFFER.C &&
X/*
X
X    Copyright (C) 1988 Free Software Foundation
X
XGNU tar is distributed in the hope that it will be useful, but WITHOUT ANY
XWARRANTY.  No author or distributor accepts responsibility to anyone
Xfor the consequences of using it or for whether it serves any
Xparticular purpose or works at all, unless he says so in writing.
XRefer to the GNU tar General Public License for full details.
X
XEveryone is granted permission to copy, modify and redistribute GNU tar,
Xbut only under the conditions described in the GNU tar General Public
XLicense.  A copy of this license is supposed to have been given to you
Xalong with GNU tar so you can know your rights and responsibilities.  It
Xshould be in a file named COPYING.  Among other things, the copyright
Xnotice and this notice must be preserved on all copies.
X
XIn other words, go ahead and share GNU tar, but don't try to stop
Xanyone else from sharing it farther.  Help stamp out software hoarding!
X*/
X
X/*
X * Buffer management for tar.
X *
X * Written by John Gilmore, ihnp4!hoptoad!gnu, on 25 August 1985.
X *
X * @(#) buffer.c 1.28 11/6/87 - gnu
X */
X
X#include <stdio.h>
X#include <errno.h>
X#ifdef atarist
```

```
X#include <types.h>
X#include <stat.h>
X#include <string.h>
X#else
X#include <sys/types.h>     /* For non-Berkeley systems */
X#include <sys/stat.h>
X#endif
X
X#include <signal.h>
X
X#ifndef MSDOS
X#include <sys/ioctl.h>
X#ifndef USG
X#include <sys/mtio.h>
X#endif
X#endif
X
X#ifdef  MSDOS
X# include <fcntl.h>
X# include <process.h>
X#else
X# ifdef XENIX
X#  include <sys/inode.h>
X# endif
X# include <sys/file.h>
X#endif
X
X#include "tar.h"
X#include "port.h"
X#include "rmt.h"
X
XFILE *msg_file;        /* Either stdout or stderr:  The thing we write messages
X                (standard msgs, not errors) to.  Stdout unless we're
X                writing a pipe, in which case stderr */
X
X#define  STDIN    0        /* Standard input  file descriptor */
X#define  STDOUT   1        /* Standard output file descriptor */
X
X#define  PREAD    0        /* Read  file descriptor from pipe() */
X#define  PWRITE   1        /* Write file descriptor from pipe() */
X
X#ifdef __GNU__
Xextern void  *malloc();
Xextern void  *valloc();
X#else
Xextern char  *malloc();
Xextern char  *valloc();
X#endif
X
Xextern char  *index(), *strcat();
Xextern char  *strcpy();
X
X/*
```

```
X * V7 doesn't have a #define for this.
X */
X#ifndef O_RDONLY
X#define  O_RDONLY  0
X#endif
X#ifndef O_RDWR
X#define O_RDWR         2
X#endif
X#ifndef O_CREAT
X#define O_CREAT        0
X#endif
X#ifndef O_BINARY
X#define O_BINARY  0
X#endif
X
X#define  MAGIC_STAT    105  /* Magic status returned by child, if
X                     it can't exec.  We hope compress/sh
X                     never return this status! */
X
Xvoid writeerror();
Xvoid readerror();
Xextern void finish_header();
X
X/*
X * The record pointed to by save_rec should not be overlaid
X * when reading in a new tape block.  Copy it to record_save_area first, and
X * change the pointer in *save_rec to point to record_save_area.
X * Saved_recno records the record number at the time of the save.
X * This is used by annofile() to print the record number of a file's
X * header record.
X */
Xstatic union record **save_rec;
X union record record_save_area;
Xstatic long      saved_recno;
X
X/*
X * PID of child program, if f_compress or remote archive access.
X */
Xstatic int   childpid = 0;
X
X/*
X * Record number of the start of this block of records
X */
Xlong    baserec;
X
X/*
X * Error recovery stuff
X */
Xstatic int   r_error_count;
X
X/*
X * Have we hit EOF yet?
X */
```

```
Xstatic int   eof;
X
X/* JF we're reading, but we just read the last record and its time to update */
Xextern time_to_start_writing;
Xint file_to_switch_to= -1; /* If remote update, close archive, and use
X                     this descriptor to write to */
X
Xstatic int volno = 1;      /* JF which volume of a multi-volume tape
X                     we're on */
X
Xchar *save_name = 0;       /* Name of the file we are currently writing */
Xlong save_totsize;         /* total size of file we are writing.  Only
X                     valid if save_name is non_zero */
Xlong save_sizeleft;        /* Where we are in the file we are writing.
X                     Only valid if save_name is non-zero */
X
Xint write_archive_to_stdout;
X
X/* Used by fl_read and fl_write to store the real info about saved names */
Xstatic char real_s_name[NAMSIZ];
Xstatic long real_s_totsize;
Xstatic long real_s_sizeleft;
X
X/*
X * Return the location of the next available input or output record.
X * Return NULL for EOF.  Once we have returned NULL, we just keep returning
X * it, to avoid accidentally going on to the next file on the "tape".
X */
Xunion record *
Xfindrec()
X{
X    if (ar_record == ar_last) {
X         if (eof)
X              return (union record *)NULL;     /* EOF */
X         flush_archive();
X         if (ar_record == ar_last) {
X              eof++;
X              return (union record *)NULL;     /* EOF */
X         }
X    }
X    return ar_record;
X}
X
X
X/*
X * Indicate that we have used all records up thru the argument.
X * (should the arg have an off-by-1? XXX FIXME)
X */
Xvoid
Xuserec(rec)
X    union record *rec;
X{
X    while(rec >= ar_record)
```

```
X         ar_record++;
X    /*
X     * Do NOT flush the archive here.  If we do, the same
X     * argument to userec() could mean the next record (if the
X     * input block is exactly one record long), which is not what
X     * is intended.
X     */
X    if (ar_record > ar_last)
X         abort();
X}
X
X
X/*
X * Return a pointer to the end of the current records buffer.
X * All the space between findrec() and endofrecs() is available
X * for filling with data, or taking data from.
X */
Xunion record *
Xendofrecs()
X{
X    return ar_last;
X}
X
X
X/*
X * Duplicate a file descriptor into a certain slot.
X * Equivalent to BSD "dup2" with error reporting.
X */
Xvoid
Xdupto(from, to, msg)
X    int from, to;
X    char *msg;
X{
X    int err;
X
X    if (from != to) {
X         (void) close(to);
X         err = dup(from);
X         if (err != to) {
X              msg_perror("cannot dup %s",msg);
X              exit(EX_SYSTEM);
X         }
X         (void) close(from);
X    }
X}
X
X#ifdef MSDOS
Xvoid
Xchild_open()
X{
X#ifdef atarist
X    fprintf(stderr,"GEMDOS %s can't use compressed or remote archives\n",tar);
X#else
```

```
X    fprintf(stderr,"MSDOS %s can't use compressed or remote archives\n",tar);
X#endif
X    exit(EX_ARGSBAD);
X}
X#else
Xvoid
Xchild_open()
X{
X    int pipe[2];
X    int err = 0;
X
X    int kidpipe[2];
X    int kidchildpid;
X
X    void ck_pipe();
X    void ck_close();
X
X#define READ 0
X#define WRITE    1
X
X    ck_pipe(pipe);
X
X    childpid=fork();
X    if(childpid<0) {
X         msg_perror("cannot fork");
X         exit(EX_SYSTEM);
X    }
X    if(childpid>0) {
X         /* We're the parent.  Clean up and be happy */
X         /* This, at least, is easy */
X
X         if(ar_reading) {
X              f_reblock++;
X              archive=pipe[READ];
X              ck_close(pipe[WRITE]);
X         } else {
X              archive = pipe[WRITE];
X              ck_close(pipe[READ]);
X         }
X         return;
X    }
X
X    /* We're the kid */
X    if(ar_reading)
X         dupto(pipe[WRITE],STDOUT,"(child) pipe to stdout");
X    else
X         dupto(pipe[READ],STDIN,"(child) pipe to stdin");
X    /* ck_close(pipe[READ]);
X    ck_close(pipe[WRITE]);*/
X
X    /* We need a child tar only if
X       1: we're reading/writing stdin/out (to force reblocking)
X       2: the file is to be accessed by rmt (compress doesn't know how
```

```
X       3: the file is not a plain file */
X#ifdef NO_REMOTE
X    if(!(ar_file[0]=='-' && ar_file[1]=='\0') && isfile(ar_file))
X#else
X    if(!(ar_file[0]=='-' && ar_file[1]=='\0') && !_remdev(ar_file) &&
isfile(ar_file))
X#endif
X    {
X         /* We don't need a child tar.  Open the archive */
X         if(ar_reading) {
X              archive=open(ar_file, O_RDONLY|O_BINARY, 0666);
X              if(archive<0) {
X                   msg_perror("can't open archive %s",ar_file);
X                   exit(EX_BADARCH);
X              }
X              dupto(archive,STDIN,"archive to stdin");
X              /* close(archive); */
X         } else {
X              archive=creat(ar_file,0666);
X              if(archive<0) {
X                   msg_perror("can't open archive %s",ar_file);
X                   exit(EX_BADARCH);
X              }
X              dupto(archive,STDOUT,"archive to stdout");
X              /* close(archive); */
X         }
X    } else {
X         /* We need a child tar */
X         ck_pipe(kidpipe);
X
X         kidchildpid=fork();
X         if(kidchildpid<0) {
X              msg_perror("child can't fork");
X              exit(EX_SYSTEM);
X         }
X
X         if(kidchildpid>0) {
X              /* About to exec compress:  set up the files */
X              if(ar_reading) {
X                   dupto(kidpipe[READ],STDIN,"((child)) pipe to stdin");
X                   ck_close(kidpipe[WRITE]);
X                   /* dup2(pipe[WRITE],STDOUT); */
X              } else {
X                   /* dup2(pipe[READ],STDIN); */
X                   dupto(kidpipe[WRITE],STDOUT,"((child)) pipe to stdout");
X                   ck_close(kidpipe[READ]);
X              }
X              /* ck_close(pipe[READ]); */
X              /* ck_close(pipe[WRITE]); */
X              /* ck_close(kidpipe[READ]);
X              ck_close(kidpipe[WRITE]); */
X         } else {
X         /* Grandchild.  Do the right thing, namely sit here and
```

```
X            read/write the archive, and feed stuff back to compress */
X             tar="tar (child)";
X             if(ar_reading) {
X                  dupto(kidpipe[WRITE],STDOUT,"[child] pipe to stdout");
X                  ck_close(kidpipe[READ]);
X             } else {
X                  dupto(kidpipe[READ],STDIN,"[child] pipe to stdin");
X                  ck_close(kidpipe[WRITE]);
X             }
X
X             if (ar_file[0] == '-' && ar_file[1] == '\0') {
X                  if (ar_reading)
X                       archive = STDIN;
X                  else
X                       archive = STDOUT;
X             } else /* This can't happen if (ar_reading==2)
X                  archive = rmtopen(ar_file, O_RDWR|O_CREAT|O_BINARY, 0666);
X             else */if(ar_reading)
X                  archive = rmtopen(ar_file, O_RDONLY|O_BINARY, 0666);
X             else
X                  archive = rmtcreat(ar_file, 0666);
X
X             if (archive < 0) {
X                  msg_perror("can't open archive %s",ar_file);
X                  exit(EX_BADARCH);
X             }
X
X             if(ar_reading) {
X                  for(;;) {
X                       char *ptr;
X                       int max,count;
X
X                       r_error_count = 0;
X                  error_loop:
X                       err=rmtread(archive, ar_block->charptr,(int)(blocksize));
X                       if(err<0) {
X                            readerror();
X                            goto error_loop;
X                       }
X                       if(err==0)
X                            break;
X                       ptr = ar_block->charptr;
X                       max = err;
X                       while(max) {
X                            count = (max<RECORDSIZE) ? max : RECORDSIZE;
X                            err=write(STDOUT,ptr,count);
X                            if(err!=count) {
X                                 if(err<0) {
X                                      msg_perror("can't write to compress");
X                                      exit(EX_SYSTEM);
X                                 } else
X                                      msg("write to compress short %d bytes",count-
err);
```

```
X                              count = (err<0) ? 0 : err;
X                         }
X                         ptr+=count;
X                         max-=count;
X                    }
X               }
X          } else {
X               for(;;) {
X                    int n;
X                    char *ptr;
X
X                    n=blocksize;
X                    ptr = ar_block->charptr;
X                    while(n) {
X                         err=read(STDIN,ptr,(n<RECORDSIZE) ? n : RECORDSIZE);
X                         if(err<=0)
X                              break;
X                         n-=err;
X                         ptr+=err;
X                    }
X                         /* EOF */
X                    if(err==0) {
X                         blocksize-=n;
X                         err=rmtwrite(archive,ar_block->charptr,blocksize);
X                         if(err!=(blocksize))
X                              writeerror(err);
X                         blocksize+=n;
X                         break;
X                    }
X                    if(n) {
X                         msg_perror("can't read from compress");
X                         exit(EX_SYSTEM);
X                    }
X                    err=rmtwrite(archive, ar_block->charptr, (int)blocksize);
X                    if(err!=blocksize)
X                         writeerror(err);
X               }
X          }
X
X          /* close_archive(); */
X          exit(0);
X     }
X }
X     /* So we should exec compress (-d) */
X if(ar_reading)
X     execlp("compress", "compress", "-d", (char *)0);
X else
X     execlp("compress", "compress", (char *)0);
X msg_perror("can't exec compress");
X _exit(EX_SYSTEM);
X}
X
X
```

```
X/* return non-zero if p is the name of a directory */
Xisfile(p)
Xchar *p;
X{
X    struct stat stbuf;
X
X    if(stat(p,&stbuf)<0)
X         return 1;
X    if((stbuf.st_mode&S_IFMT)==S_IFREG)
X         return 1;
X    return 0;
X}
X
X#endif
X
X#ifdef DONTDEF
X/*
X * Fork a child to deal with remote files or compression.
X * If rem_host is zero, we are called only for compression.
X */
Xvoid
Xchild_open(rem_host, rem_file)
X    char *rem_host, *rem_file;
X{
X
X#ifdef  MSDOS
X    fprintf(stderr,
X#ifdef atarist
X      "MSDOS %s cannot deal with compressed or remote archives\n", tar);
X#else
X      "MSDOS %s cannot deal with compressed or remote archives\n", tar);
X#endif
X    exit(EX_ARGSBAD);
X#else
X
X    int pipes[2];
X    int err;
X    struct stat arstat;
X    char cmdbuf[1000];      /* For big file and host names */
X
X    int other_pipes[2];     /* JF for remote update and Multivol */
X
X    /* Create a pipe to talk to the child over */
X    err = pipe(pipes);
X    if (err < 0) {
X         msg_perror ("cannot create pipe to child");
X         exit(EX_SYSTEM);
X    }
X
X    if(cmd_mode==CMD_CAT || cmd_mode==CMD_UPDATE || cmd_mode==CMD_APPEND) {
X         err=pipe(other_pipes);
X         if(err<0) {
X              msg_perror("cannot create a pipe");
```

```
X             exit(EX_SYSTEM);
X         }
X    }
X
X    /* Fork child process */
X    childpid = fork();
X    if (childpid < 0) {
X         msg_perror("cannot fork");
X         exit(EX_SYSTEM);
X    }
X
X    /*
X     * Parent process.  Clean up.
X     *
X     * We always close the archive file (stdin, stdout, or opened file)
X     * since the child will end up reading or writing that for us.
X     * Note that this may leave standard input closed.
X     * We close the child's end of the pipe since they will handle
X     * that too; and we set <archive> to the other end of the pipe.
X     *
X     * If reading, we set f_reblock since reading pipes or network
X     * sockets produces odd length data.
X     */
X    if (childpid > 0) {
X         if(f_multivol || cmd_mode==CMD_CAT || cmd_mode==CMD_UPDATE ||
cmd_mode==CMD_APPEND) {
X              (void)close(other_pipes[PREAD]);
X              file_to_switch_to=other_pipes[PWRITE];
X         }
X         (void) close (archive);
X         if (ar_reading) {
X              (void) close (pipes[PWRITE]);
X              archive = pipes[PREAD];
X              f_reblock++;
X         } else {
X              (void) close (pipes[PREAD]);
X              archive = pipes[PWRITE];
X         }
X         return;
X    }
X
X    /*
X     * Child process.
X     */
X
X    /* We can't do any of these to a compressed file, so it'd better
X       be remote */
X    if(f_multivol || cmd_mode==CMD_UPDATE || cmd_mode==CMD_APPEND ||
cmd_mode==CMD_CAT) {
X         char mode;
X
X         (void) close(pipes[PREAD]);
X         dupto(pipes[PWRITE],STDOUT,"(child) to stdout");
```

```
X         (void) close(other_pipes[PWRITE]);
X         dupto(other_pipes[PREAD],STDIN,"(child) to stdin");
X         if(f_multivol)  {
X              if(cmd_mode==CMD_CREATE)
X                   mode='c';
X              else if(cmd_mode==CMD_DIFF || cmd_mode==CMD_LIST ||
cmd_mode==CMD_EXTRACT)
X                   mode='r';
X              else mode='c';
X         } else
X              mode='c';
X
X         (void) sprintf(cmdbuf,"rsh '%s' tar -S%c -b %d -f
'%s'",rem_host,mode,blocking,rem_file);
X
X         execlp("sh","sh","-c",cmdbuf,(char *)0);
X         msg_perror("cannot exec sh");
X         exit(MAGIC_STAT);
X    }
X
X    if (ar_reading) {
X         /*
X          * Reading from the child...
X          *
X          * Close the read-side of the pipe, which our parent will use.
X          * Move the write-side of pipe to stdout,
X          * If local, move archive input to child's stdin,
X          * then run the child.
X          */
X         (void) close (pipes[PREAD]);
X         dupto(pipes[PWRITE], STDOUT, "to stdout");
X         if (rem_host) {
X              (void) close (STDIN);  /* rsh abuses stdin */
X              if (STDIN != open("/dev/null"))
X                   msg_perror("Can't open /dev/null");
X              (void) sprintf(cmdbuf,
X                   "rsh '%s' dd '<%s' bs=%db",
X                   rem_host, rem_file, blocking);
X              if (f_compress)
X                   strcat(cmdbuf, "| compress -d");
X#ifdef DEBUG
X              fprintf(stderr, "Exec-ing: %s\n", cmdbuf);
X#endif
X              execlp("sh", "sh", "-c", cmdbuf, (char *)0);
X              msg_perror("cannot exec sh");
X         } else {
X              /*
X               * If we are reading a disk file, compress is OK;
X               * otherwise, we have to reblock the input in case it's
X               * coming from a tape drive.  This is an optimization.
X               */
X              dupto(archive, STDIN, "to stdin");
X              err = fstat(STDIN, &arstat);
```

```
X             if (err != 0) {
X                  msg_perror("can't fstat archive");
X                  exit(EX_SYSTEM);
X             }
X             if ((arstat.st_mode & S_IFMT) == S_IFREG) {
X                  execlp("compress", "compress", "-d", (char *)0);
X                  msg_perror("cannot exec compress");
X             } else {
X                  /* Non-regular file needs dd before compress */
X                  (void) sprintf(cmdbuf,
X                       "dd bs=%db | compress -d",
X                       blocking);
X#ifdef DEBUG
X                  fprintf(stderr, "Exec-ing: %s\n", cmdbuf);
X#endif
X                  execlp("sh", "sh", "-c", cmdbuf, (char *)0);
X                  msg_perror("cannot exec sh");
X             }
X        }
X        exit(MAGIC_STAT);
X    } else {
X         /*
X          * Writing archive to the child.
X          * It would like to run either:
X          *   compress
X          *   compress |            dd obs=20b
X          *            rsh 'host' dd obs=20b '>foo'
X          * or    compress | rsh 'host' dd obs=20b '>foo'
X          *
X          * We need the dd to reblock the output to the
X          * user's specs, if writing to a device or over
X          * the net.  However, it produces a stupid
X          * message about how many blocks it processed.
X          * Because the shell on the remote end could be just
X          * about any shell, we can't depend on it to do
X          * redirect stderr properly for us -- the csh
X          * doesn't use the same syntax as the Bourne shell.
X          * On the other hand, if we just ignore stderr on
X          * this end, we won't see errors from rsh, or from
X          * the inability of "dd" to write its output file.
X          * The combination of the local sh, the rsh, the
X          * remote csh, and maybe a remote sh conspires to mess
X          * up any possible quoting method, so grumble! we
X          * punt and just accept the fucking "xxx blocks"
X          * messages.  The real fix would be a "dd" that
X          * would shut up.
X          *
X          * Close the write-side of the pipe, which our parent will use.
X          * Move the read-side of the pipe to stdin,
X          * If local, move archive output to the child's stdout.
X          * then run the child.
X          */
X         (void) close (pipes[PWRITE]);
```

```
X         dupto(pipes[PREAD], STDIN, "to stdin");
X         if (!rem_host)
X              dupto(archive, STDOUT, "to stdout");
X
X         cmdbuf[0] = '\0';
X         if (f_compress) {
X              if (!rem_host) {
X                   err = fstat(STDOUT, &arstat);
X                   if (err != 0) {
X                        msg_perror("can't fstat archive");
X                        exit(EX_SYSTEM);
X                   }
X                   if ((arstat.st_mode & S_IFMT) == S_IFREG) {
X                        execlp("compress", "compress", (char *)0);
X                        msg_perror("cannot exec compress");
X                        _exit(EX_SYSTEM);
X                   }
X              }
X              strcat(cmdbuf, "compress | ");
X         }
X         if (rem_host) {
X              (void) sprintf(cmdbuf+strlen(cmdbuf),
X                "rsh '%s' dd obs=%db '>%s'",
X                   rem_host, blocking, rem_file);
X         } else {
X              (void) sprintf(cmdbuf+strlen(cmdbuf),
X                   "dd obs=%db",
X                   blocking);
X         }
X#ifdef DEBUG
X         fprintf(stderr, "Exec-ing: %s\n", cmdbuf);
X#endif
X         execlp("sh", "sh", "-c", cmdbuf, (char *)0);
X         msg_perror("cannot exec sh");
X         exit(MAGIC_STAT);
X    }
X#endif  /* MSDOS */
X}
X
X#endif
X
X/*
X * Open an archive file.  The argument specifies whether we are
X * reading or writing.
X */
X/* JF if the arg is 2, open for reading and writing. */
Xopen_archive(reading)
X    int reading;
X{
X    msg_file = stdout;
X
X/* #ifndef MSDOS */
X#ifdef DonTDeF
```

```
X    /* JF Not doing this on MSdos systems looks like a good idea.
X       Does this stuff about 'slash' mean I can't simply say
X       remotesys:iggy to store it in 'iggy' on remotesys?  Do I
X       have to say remotesys:./iggy in order for it to work?
X       Who knows??  */
X    colon = index(ar_file, ':');
X    if (colon) {
X         slash = index(ar_file, '/');
X         if (slash && slash > colon) {
X              /*
X               * Remote file specified.  Parse out separately,
X               * and don't try to open it on the local system.
X               */
X              rem_file = colon + 1;
X              rem_host = ar_file;
X              *colon = '\0';
X              goto gotit;
X         }
X    }
X#endif
X
X    if (blocksize == 0) {
X         msg("invalid value for blocksize");
X         exit(EX_ARGSBAD);
X    }
X
X    /*NOSTRICT*/
X    if(f_multivol) {
X         ar_block = (union record *) valloc((unsigned)(blocksize+(2*RECORDSIZE)));
X         if(ar_block)
X              ar_block += 2;
X    } else
X         ar_block = (union record *) valloc((unsigned)blocksize);
X    if (!ar_block) {
X         msg("could not allocate memory for blocking factor %d",
X              blocking);
X         exit(EX_ARGSBAD);
X    }
X
X    ar_record = ar_block;
X    ar_last   = ar_block + blocking;
X    ar_reading = reading;
X
X    if (f_compress) {
X         if(reading==2 || f_verify) {
X              msg("cannot update or verify compressed archives");
X              exit(EX_ARGSBAD);
X         }
X         child_open();
X         if(!reading && ar_file[0]=='-' && ar_file[1]=='\0')
X              msg_file = stderr;
X         /* child_open(rem_host, rem_file); */
X    } else if (ar_file[0] == '-' && ar_file[1] == '\0') {
```

```
X         f_reblock++;  /* Could be a pipe, be safe */
X         if(f_verify) {
X              msg("can't verify stdin/stdout archive");
X              exit(EX_ARGSBAD);
X         }
X         if(reading==2) {
X              archive=STDIN;
X              msg_file=stderr;
X              write_archive_to_stdout++;
X         } else if (reading)
X              archive = STDIN;
X         else {
X              archive = STDOUT;
X              msg_file = stderr;
X         }
X    } else if (reading==2 || f_verify) {
X         archive = rmtopen(ar_file, O_RDWR|O_CREAT|O_BINARY, 0666);
X    } else if(reading) {
X         archive = rmtopen(ar_file, O_RDONLY|O_BINARY, 0666);
X    } else {
X         archive = rmtcreat(ar_file, 0666);
X    }
X
X    if (archive < 0) {
X         msg_perror("can't open %s",ar_file);
X         exit(EX_BADARCH);
X    }
X#ifdef  MSDOS
X    setmode(archive, O_BINARY);
X#endif
X
X    if (reading) {
X         ar_last = ar_block;         /* Set up for 1st block = # 0 */
X         (void) findrec();      /* Read it in, check for EOF */
X
X         if(f_volhdr) {
X              union record *head;
X              char *ptr;
X
X              if(f_multivol) {
X                   ptr=malloc(strlen(f_volhdr)+20);
X                   sprintf(ptr,"%s Volume %d",f_volhdr,1);
X              } else
X                   ptr=f_volhdr;
X              head=findrec();
X              if(!head)
X                   exit(EX_BADARCH);
X              if(strcmp(ptr,head->header.name)) {
X                   msg("Volume mismatch!  %s!=%s\n",ptr,head->header.name);
X                   exit(EX_BADARCH);
X              }
X              if(ptr!=f_volhdr)
X                   free(ptr);
```

```
X         }
X    } else if(f_volhdr) {
X         bzero((void *)ar_block,RECORDSIZE);
X         if(f_multivol)
X              sprintf(ar_block->header.name,"%s Volume 1",f_volhdr);
X         else
X              strcpy(ar_block->header.name,f_volhdr);
X         ar_block->header.linkflag = LF_VOLHDR;
X         finish_header(ar_block);
X         /* ar_record++; */
X    }
X}
X
X
X/*
X * Remember a union record * as pointing to something that we
X * need to keep when reading onward in the file.  Only one such
X * thing can be remembered at once, and it only works when reading
X * an archive.
X *
X * We calculate "offset" then add it because some compilers end up
X * adding (baserec+ar_record), doing a 9-bit shift of baserec, then
X * subtracting ar_block from that, shifting it back, losing the top 9 bits.
X */
Xsaverec(pointer)
X    union record **pointer;
X{
X    long offset;
X
X    save_rec = pointer;
X    offset = ar_record - ar_block;
X    saved_recno = baserec + offset;
X}
X
X/*
X * Perform a write to flush the buffer.
X */
X
X/*send_buffer_to_file();
X  if(new_volume) {
X    deal_with_new_volume_stuff();
X    send_buffer_to_file();
X  }
X */
X
Xfl_write()
X{
X    int err;
X    int copy_back;
X    extern int errno;
X#ifdef TEST
X    static long test_written = 0;
X#endif
```

```
X
X#ifdef TEST
X    if(test_written>=30720) {
X         errno = ENOSPC;
X         err = 0;
X    } else
X#endif
X    err = rmtwrite(archive, ar_block->charptr,(int) blocksize);
X    if(err!=blocksize && !f_multivol)
X         writeerror(err);
X
X#ifdef TEST
X    if(err>0)
X         test_written+=err;
X#endif
X    if (err == blocksize) {
X         if(f_multivol) {
X              if(!save_name) {
X                   real_s_name[0]='\0';
X                   real_s_totsize=0;
X                   real_s_sizeleft = 0;
X                   return;
X              }
X#ifdef MSDOS
X              if(save_name[1]==':')
X                   save_name+=2;
X#endif
X              while(*save_name=='/')
X                   save_name++;
X
X              strcpy(real_s_name,save_name);
X              real_s_totsize = save_totsize;
X              real_s_sizeleft = save_sizeleft;
X         }
X         return;
X    }
X
X#ifndef atarist /* ENOSPC isn't available under GEMDOS; punt */
X    /* We're multivol  Panic if we didn't get the right kind of response */
X    if(err>0 || (errno!=ENOSPC && errno!=EIO))
X         writeerror(err);
X#endif
X
X    if(new_volume(0)<0)
X         return;
X#ifdef TEST
X    test_written=0;
X#endif
X    if(f_volhdr && real_s_name[0]) {
X         copy_back=2;
X         ar_block-=2;
X    } else if(f_volhdr || real_s_name[0]) {
X         copy_back = 1;
```

```
X         ar_block--;
X    } else
X         copy_back = 0;
X    if(f_volhdr) {
X         bzero((void *)ar_block,RECORDSIZE);
X         sprintf(ar_block->header.name,"%s Volume %d",f_volhdr,volno);
X         ar_block->header.linkflag = LF_VOLHDR;
X         finish_header(ar_block);
X    }
X    if(real_s_name[0]) {
X         extern void to_oct();
X         int tmp;
X
X         if(f_volhdr)
X              ar_block++;
X         bzero((void *)ar_block,RECORDSIZE);
X         strcpy(ar_block->header.name,real_s_name);
X         ar_block->header.linkflag = LF_MULTIVOL;
X         to_oct((long)real_s_sizeleft,1+12,
X                ar_block->header.size);
X         to_oct((long)real_s_totsize-real_s_sizeleft,
X                1+12,ar_block->header.offset);
X         tmp=f_verbose;
X         f_verbose=0;
X         finish_header(ar_block);
X         f_verbose=tmp;
X         if(f_volhdr)
X              ar_block--;
X    }
X
X    err = rmtwrite(archive, ar_block->charptr,(int) blocksize);
X    if(err!=blocksize)
X         writeerror(err);
X
X#ifdef TEST
X    test_written = blocksize;
X#endif
X    if(copy_back) {
X         ar_block+=copy_back;
X         bcopy((void *)(ar_block+blocking-copy_back),
X               (void *)ar_record,
X               copy_back*RECORDSIZE);
X         ar_record+=copy_back;
X
X         if(real_s_sizeleft>=copy_back*RECORDSIZE)
X              real_s_sizeleft-=copy_back*RECORDSIZE;
X         else if((real_s_sizeleft+RECORDSIZE-1)/RECORDSIZE<=copy_back)
X              real_s_name[0] = '\0';
X         else {
X#ifdef MSDOS
X              if(save_name[1]==':')
X                   save_name+=2;
X#endif
```

```
X             while(*save_name=='/')
X                  save_name++;
X
X             strcpy(real_s_name,save_name);
X             real_s_sizeleft = save_sizeleft;
X             real_s_totsize=save_totsize;
X         }
X         copy_back = 0;
X    }
X}
X
X/* Handle write errors on the archive.  Write errors are always fatal */
X/* Hitting the end of a volume does not cause a write error unless the write
X*  was the first block of the volume */
X
Xvoid
Xwriteerror(err)
Xint err;
X{
X    if (err < 0) {
X         msg_perror("can't write to %s",ar_file);
X         exit(EX_BADARCH);
X    } else {
X         msg("write to %s short %d bytes",ar_file,blocksize-err);
X         exit(EX_BADARCH);
X    }
X}
X
X/*
X * Handle read errors on the archive.
X *
X * If the read should be retried, readerror() returns to the caller.
X */
Xvoid
Xreaderror()
X{
X#   define    READ_ERROR_MAX     10
X
X    read_error_flag++;      /* Tell callers */
X
X    msg_perror("read error on %s",ar_file);
X
X    if (baserec == 0) {
X         /* First block of tape.  Probably stupidity error */
X         exit(EX_BADARCH);
X    }
X
X    /*
X     * Read error in mid archive.  We retry up to READ_ERROR_MAX times
X     * and then give up on reading the archive.  We set read_error_flag
X     * for our callers, so they can cope if they want.
X     */
X    if (r_error_count++ > READ_ERROR_MAX) {
```

```
X         msg("Too many errors, quitting.");
X         exit(EX_BADARCH);
X    }
X    return;
X}
X
X
X/*
X * Perform a read to flush the buffer.
X */
Xfl_read()
X{
X    int err;      /* Result from system call */
X    int left;     /* Bytes left */
X    char *more;        /* Pointer to next byte to read */
X
X    /*
X     * Clear the count of errors.  This only applies to a single
X     * call to fl_read.  We leave read_error_flag alone; it is
X     * only turned off by higher level software.
X     */
X    r_error_count = 0; /* Clear error count */
X
X    /*
X     * If we are about to wipe out a record that
X     * somebody needs to keep, copy it out to a holding
X     * area and adjust somebody's pointer to it.
X     */
X    if (save_rec &&
X        *save_rec >= ar_record &&
X        *save_rec < ar_last) {
X         record_save_area = **save_rec;
X         *save_rec = &record_save_area;
X    }
X    if(write_archive_to_stdout && baserec!=0) {
X         err=rmtwrite(1, ar_block->charptr, blocksize);
X         if(err!=blocksize)
X              writeerror(err);
X    }
X    if(f_multivol) {
X         if(save_name) {
X              if(save_name!=real_s_name) {
X#ifdef MSDOS
X                   if(save_name[1]==':')
X                        save_name+=2;
X#endif
X                   while(*save_name=='/')
X                        save_name++;
X
X                   strcpy(real_s_name,save_name);
X                   save_name=real_s_name;
X              }
X              real_s_totsize = save_totsize;
```

```
X             real_s_sizeleft = save_sizeleft;
X
X        } else {
X             real_s_name[0]='\0';
X             real_s_totsize=0;
X             real_s_sizeleft = 0;
X        }
X    }
X
Xerror_loop:
X    err = rmtread(archive, ar_block->charptr, (int)blocksize);
X    if (err == blocksize)
X         return;
X
X    if (err < 0) {
X         readerror();
X         goto error_loop;   /* Try again */
X    }
X
X    if(err == 0 && f_multivol) {
X         union record *head;
X
X    try_volume:
X         if(new_volume((cmd_mode==CMD_APPEND || cmd_mode==CMD_CAT ||
cmd_mode==CMD_UPDATE) ? 2 : 1)<0)
X              return;
X    vol_error:
X         err = rmtread(archive, ar_block->charptr,(int) blocksize);
X         if(err < 0) {
X              readerror();
X              goto vol_error;
X         }
X         if(err!=blocksize)
X              goto short_read;
X
X         head=ar_block;
X
X         if(head->header.linkflag==LF_VOLHDR) {
X              if(f_volhdr) {
X                   char *ptr;
X
X                   ptr=(char *)malloc(strlen(f_volhdr)+20);
X                   sprintf(ptr,"%s Volume %d",f_volhdr,volno);
X                   if(strcmp(ptr,head->header.name)) {
X                        msg("Volume mismatch! %s!=%s\n",ptr,head->header.name);
X                        --volno;
X                        free(ptr);
X                        goto try_volume;
X                   }
X                   free(ptr);
X              }
X              if(f_verbose)
X                   fprintf(msg_file,"Reading %s\n",head->header.name);
```

```
X             head++;
X        } else if(f_volhdr) {
X             msg("Warning:  No volume header!");
X        }
X
X        if(real_s_name[0]) {
X             long from_oct();
X
X             if(head->header.linkflag!=LF_MULTIVOL || strcmp(head-
>header.name,real_s_name)) {
X                  msg("%s is not continued on this volume!",real_s_name);
X                  --volno;
X                  goto try_volume;
X             }
X             if(real_s_totsize!=from_oct(1+12,head->header.size)
+from_oct(1+12,head->header.offset)) {
X                  msg("%s is the wrong size (%ld!=%ld+%ld)",
X                        head->header.name,save_totsize,
X                        from_oct(1+12,head->header.size),
X                        from_oct(1+12,head->header.offset));
X                  --volno;
X                  goto try_volume;
X             }
X             if(real_s_totsize-real_s_sizeleft!=from_oct(1+12,head-
>header.offset)) {
X                  msg("This volume is out of sequence");
X                  --volno;
X                  goto try_volume;
X             }
X             head++;
X        }
X        ar_record=head;
X        return;
X   }
X
X short_read:
X   more = ar_block->charptr + err;
X   left = blocksize - err;
X
Xagain:
X   if (0 == (((unsigned)left) % RECORDSIZE)) {
X        /* FIXME, for size=0, multi vol support */
X        /* On the first block, warn about the problem */
X        if (!f_reblock && baserec == 0 && f_verbose && err > 0) {
X             msg("Blocksize = %d record%s",
X                  err / RECORDSIZE, (err > RECORDSIZE)? "s": "");
X        }
X        ar_last = ar_block + ((unsigned)(blocksize - left))/RECORDSIZE;
X        return;
X   }
X   if (f_reblock) {
X        /*
X         * User warned us about this.  Fix up.
```

```
X         */
X        if (left > 0) {
Xerror2loop:
X             err = rmtread(archive, more, (int)left);
X             if (err < 0) {
X                  readerror();
X                  goto error2loop;   /* Try again */
X             }
X             if (err == 0) {
X                  msg("archive %s EOF not no block boundry",ar_file);
X                  exit(EX_BADARCH);
X             }
X             left -= err;
X             more += err;
X             goto again;
X        }
X    } else {
X         msg("only read %d byes from archive %s",err,ar_file);
X         exit(EX_BADARCH);
X    }
X}
X
X
X/*
X * Flush the current buffer to/from the archive.
X */
Xflush_archive()
X{
X    baserec += ar_last - ar_block;   /* Keep track of block #s */
X    ar_record = ar_block;       /* Restore pointer to start */
X    ar_last = ar_block + blocking;   /* Restore pointer to end */
X
X    if (ar_reading) {
X         if(time_to_start_writing) {
X              time_to_start_writing=0;
X              ar_reading=0;
X
X              if(file_to_switch_to>=0) {
X                   rmtclose(archive);
X                   archive=file_to_switch_to;
X              } else
X                   (void)backspace_output();
X              fl_write();
X         } else
X              fl_read();
X    } else {
X         fl_write();
X    }
X}
X
X/* Backspace the archive descriptor by one blocks worth.
X   If its a tape, MTIOCTOP will work.  If its something else,
X   we try to seek on it.  If we can't seek, we lose! */
```

```
Xbackspace_output()
X{
X    long cur;
X    /* int er; */
X    extern char *output_start;
X    extern int errno;
X
X#ifdef MTIOCTOP
X    struct mtop t;
X
X    t.mt_op = MTBSR;
X    t.mt_count = 1;
X    if((rmtioctl(archive,MTIOCTOP,&t))>=0)
X         return 1;
X    if(errno==EIO && (rmtioctl(archive,MTIOCTOP,&t))>=0)
X         return 1;
X#endif
X
X         cur=rmtlseek(archive,0L,1);
X    cur-=blocksize;
X    /* Seek back to the beginning of this block and
X       start writing there. */
X
X    if(rmtlseek(archive,cur,0)!=cur) {
X         /* Lseek failed.  Try a different method */
X         msg("Couldn't backspace archive file.  It may be unreadable without
-i.");
X         /* Replace the first part of the block with nulls */
X         if(ar_block->charptr!=output_start)
X              bzero(ar_block->charptr,output_start-ar_block->charptr);
X         return 2;
X    }
X    return 3;
X}
X
X
X/*
X * Close the archive file.
X */
Xclose_archive()
X{
X    int child;
X    int status;
X
X    if (time_to_start_writing || !ar_reading)
X         flush_archive();
X    if(cmd_mode==CMD_DELETE) {
X         long pos;
X
X         pos = rmtlseek(archive,0L,1);
X#ifndef MSDOS
X         /* FIXME does ftruncate really take an INT?! */
X         (void) ftruncate(archive,(int)pos);
```

```
X#else
X        (void)rmtwrite(archive,"",0);
X#endif
X    }
X    if(f_verify)
X         verify_volume();
X    (void) rmtclose(archive);
X
X#ifndef MSDOS
X    if (childpid) {
X         /*
X          * Loop waiting for the right child to die, or for
X          * no more kids.
X          */
X         while (((child = wait(&status)) != childpid) && child != -1)
X              ;
X
X         if (child != -1) {
X              switch (TERM_SIGNAL(status)) {
X              case 0:
X                   /* Child voluntarily terminated  -- but why? */
X                   if (TERM_VALUE(status) == MAGIC_STAT) {
X                        exit(EX_SYSTEM);/* Child had trouble */
X                   }
X                   if (TERM_VALUE(status) == (SIGPIPE + 128)) {
X                        /*
X                         * /bin/sh returns this if its child
X                         * dies with SIGPIPE.  'Sok.
X                         */
X                        break;
X                   } else if (TERM_VALUE(status))
X                        msg("child returned status %d",
X                            TERM_VALUE(status));
X              case SIGPIPE:
X                   break;        /* This is OK. */
X
X              default:
X                   msg("child died with signal %d%s",
X                    TERM_SIGNAL(status),
X                    TERM_COREDUMP(status)? " (core dumped)": "");
X              }
X         }
X    }
X#endif  /* MSDOS */
X}
X
X
X#ifdef DONTDEF
X/*
X * Message management.
X *
X * anno writes a message prefix on stream (eg stdout, stderr).
X *
```

```
X * The specified prefix is normally output followed by a colon and a space.
X * However, if other command line options are set, more output can come
X * out, such as the record # within the archive.
X *
X * If the specified prefix is NULL, no output is produced unless the
X * command line option(s) are set.
X *
X * If the third argument is 1, the "saved" record # is used; if 0, the
X * "current" record # is used.
X */
Xvoid
Xanno(stream, prefix, savedp)
X    FILE *stream;
X    char *prefix;
X    int  savedp;
X{
X#   define    MAXANNO   50
X    char buffer[MAXANNO];   /* Holds annorecment */
X#   define    ANNOWIDTH 13
X    int  space;
X    long offset;
X    int  save_e;
X
X    save_e=errno;
X    /* Make sure previous output gets out in sequence */
X    if (stream == stderr)
X         fflush(stdout);
X    if (f_sayblock) {
X         if (prefix) {
X              fputs(prefix, stream);
X              putc(' ', stream);
X         }
X         offset = ar_record - ar_block;
X         (void) sprintf(buffer, "rec %d: ",
X              savedp?  saved_recno:
X                   baserec + offset);
X         fputs(buffer, stream);
X         space = ANNOWIDTH - strlen(buffer);
X         if (space > 0) {
X              fprintf(stream, "%*s", space, "");
X         }
X    } else if (prefix) {
X         fputs(prefix, stream);
X         fputs(": ", stream);
X    }
X    errno=save_e;
X}
X#endif
X
X/* We've hit the end of the old volume.  Close it and open the next one */
X/* Values for type:  0: writing  1: reading  2: updating */
Xnew_volume(type)
Xint type;
```

```
X{
X    /* int   c; */
X    char inbuf[80];
X    char *p;
X    extern int now_verifying;
X    char *getenv();
X
X    if(now_verifying)
X         return -1;
X    if(f_verify)
X         verify_volume();
X    if(rmtclose(archive)<0) {
X         msg_perror("can't close %s",ar_file);
X         exit(EX_BADARCH);
X    }
X    volno++;
X    for(;;) {
X         fprintf(msg_file,"Prepare volume #%d and hit return: ",volno);
X         if(fgets(inbuf,80,stdin)==0 || inbuf[0]=='\n')
X              break;
X         switch(inbuf[0]) {
X         case '?':
X              fprintf(msg_file,"\
X n [name]   Give a new filename for the next (and subsequent) volume(s)\n\
X q          Abort tar\n\
X !          Spawn a subshell\n\
X ?          Print this list\n");
X              break;
X
X         case 'q':/* Quit */
X              fprintf(msg_file,"No new volume; exiting.\n");
X              if(cmd_mode!=CMD_EXTRACT && cmd_mode!=CMD_LIST && cmd_mode!
=CMD_DIFF)
X                   msg("Warning:  Archive is INCOMPLETE!");
X              exit(EX_BADARCH);
X
X         case 'n':/* Get new file name */
X         {
X              char *q,*r;
X              static char *old_name;
X
X              for(q= &inbuf[1];*q==' ' || *q=='\t';q++)
X                   ;
X              for(r=q;*r;r++)
X                   if(*r=='\n')
X                        *r='\0';
X              if(old_name)
X                   free(old_name);
X              old_name=p=(char *)malloc((unsigned)(strlen(q)+2));
X              if(p==0) {
X                   msg("Can't allocate memory for name");
X                   exit(EX_SYSTEM);
X              }
```

```
X             (void) strcpy(p,q);
X             ar_file=p;
X             break;
X         }
X
X         case '!':
X#ifdef MSDOS
X# ifdef atarist
X             spawnl(P_WAIT,getenv("SHELL"),NULL);
X# else
X             spawnl(P_WAIT,getenv("COMSPEC"),"-",0);
X# endif
X#else
X                  /* JF this needs work! */
X             switch(fork()) {
X             case -1:
X                  msg_perror("can't fork!");
X                  break;
X             case 0:
X                  p=getenv("SHELL");
X                  if(p==0) p="/bin/sh";
X                  execlp(p,"-sh","-i",0);
X                  msg_perror("can't exec a shell %s",p);
X                  _exit(55);
X             default:
X                  wait(0);
X                  break;
X             }
X#endif
X             break;
X         }
X    }
X
X    if(type==2 || f_verify)
X         archive=rmtopen(ar_file,O_RDWR|O_CREAT,0666);
X    else if(type==1)
X         archive=rmtopen(ar_file,O_RDONLY,0666);
X    else if(type==0)
X         archive=rmtcreat(ar_file,0666);
X    else
X         archive= -1;
X
X    if(archive<0) {
X         msg_perror("can't open %s",ar_file);
X         exit(EX_BADARCH);
X    }
X#ifdef MSDOS
X    setmode(archive,O_BINARY);
X#endif
X    return 0;
X}
X
X/* this is a useless function that takes a buffer returned by wantbytes
```

```
X   and does nothing with it.  If the function called by wantbytes returns
X   an error indicator (non-zero), this function is called for the rest of
X   the file.
X */
Xint
Xno_op(size,data)
Xint size;
Xchar *data;
X{
X    return 0;
X}
X
X/* Some other routine wants SIZE bytes in the archive.  For each chunk of
X   the archive, call FUNC with the size of the chunk, and the address of
X   the chunk it can work with.
X */
Xint
Xwantbytes(size,func)
Xlong size;
Xint (*func)();
X{
X    char *data;
X    long data_size;
X
X    while(size) {
X         data = findrec()->charptr;
X         if (data == NULL) {    /* Check it... */
X              msg("Unexpected EOF on archive file");
X              return -1;
X         }
X         data_size = endofrecs()->charptr - data;
X         if(data_size>size)
X              data_size=size;
X         if((*func)(data_size,data))
X              func=no_op;
X         userec((union record *)(data + data_size - 1));
X         size-=data_size;
X    }
X    return 0;
X}
SHAR_EOF
chmod 0644 BUFFER.C || echo "restore of BUFFER.C fails"
echo "x - extracting COPYING (Text)"
sed 's/^X//' << 'SHAR_EOF' > COPYING &&
X
X            GNU TAR GENERAL PUBLIC LICENSE
X            (Clarified 11 Feb 1988)
X
X Copyright (C) 1988 Richard M. Stallman
X Everyone is permitted to copy and distribute verbatim copies
X of this license, but changing it is not allowed.  You can also
X use this wording to make the terms for other programs.
X
```

X The license agreements of most software companies keep you at the
Xmercy of those companies. By contrast, our general public license is
Xintended to give everyone the right to share GNU tar. To make sure that
Xyou get the rights we want you to have, we need to make restrictions
Xthat forbid anyone to deny you these rights or to ask you to surrender
Xthe rights. Hence this license agreement.
X
X Specifically, we want to make sure that you have the right to give
Xaway copies of GNU tar, that you receive source code or else can get it
Xif you want it, that you can change GNU tar or use pieces of it in new
Xfree programs, and that you know you can do these things.
X
X To make sure that everyone has such rights, we have to forbid you to
Xdeprive anyone else of these rights. For example, if you distribute
Xcopies of GNU tar, you must give the recipients all the rights that you
Xhave. You must make sure that they, too, receive or can get the
Xsource code. And you must tell them their rights.
X
X Also, for our own protection, we must make certain that everyone
Xfinds out that there is no warranty for GNU tar. If GNU tar is modified by
Xsomeone else and passed on, we want its recipients to know that what
Xthey have is not what we distributed, so that any problems introduced
Xby others will not reflect on our reputation.
X
X Therefore we (Richard Stallman and the Free Software Foundation,
XInc.) make the following terms which say what you must do to be
Xallowed to distribute or change GNU tar.
X
X
X   COPYING POLICIES
X
X 1. You may copy and distribute verbatim copies of GNU tar source code as
Xyou receive it, in any medium, provided that you conspicuously and
Xappropriately publish on each copy a valid copyright notice "Copyright
X(C) 1988 Free Software Foundation, Inc." (or with whatever year is
Xappropriate); keep intact the notices on all files that refer to this
XLicense Agreement and to the absence of any warranty; and give any
Xother recipients of the GNU tar program a copy of this License
XAgreement along with the program. You may charge a distribution
Xfee for the physical act of transferring a copy.
X
X 2. You may modify your copy or copies of GNU tar or any portion of it,
Xand copy and distribute such modifications under the terms of
XParagraph 1 above, provided that you also do the following:
X
X   a) cause the modified files to carry prominent notices stating
X   that you changed the files and the date of any change; and
X
X   b) cause the whole of any work that you distribute or publish,
X   that in whole or in part contains or is a derivative of GNU tar or
X   any part thereof, to be licensed at no charge to all third
X   parties on terms identical to those contained in this License
X   Agreement (except that you may choose to grant more extensive

X   warranty protection to third parties, at your option).
X
X   c) You may charge a distribution fee for the physical act of
X   transferring a copy, and you may at your option offer warranty
X   protection in exchange for a fee.
X
XMere aggregation of another unrelated program with this program (or its
Xderivative) on a volume of a storage or distribution medium does not bring
Xthe other program under the scope of these terms.
X
X 3. You may copy and distribute GNU CC (or a portion or derivative of it,
Xunder Paragraph 2) in object code or executable form under the terms of
XParagraphs 1 and 2 above provided that you also do one of the following:
X
X   a) accompany it with the complete corresponding machine-readable
X   source code, which must be distributed under the terms of
X   Paragraphs 1 and 2 above; or,
X
X   b) accompany it with a written offer, valid for at least three
X   years, to give any third party free (except for a nominal
X   shipping charge) a complete machine-readable copy of the
X   corresponding source code, to be distributed under the terms of
X   Paragraphs 1 and 2 above; or,
X
X   c) accompany it with the information you received as to where the
X   corresponding source code may be obtained. (This alternative is
X   allowed only for noncommercial distribution and only if you
X   received the program in object code or executable form alone.)
X
XFor an executable file, complete source code means all the source code for
Xall modules it contains; but, as a special exception, it need not include
Xsource code for modules which are standard libraries that accompany the
Xoperating system on which the executable file runs.
X
X 4. You may not copy, sublicense, distribute or transfer GNU tar
Xexcept as expressly provided under this License Agreement. Any attempt
Xotherwise to copy, sublicense, distribute or transfer GNU tar is void and
Xyour rights to use the program under this License agreement shall be
Xautomatically terminated. However, parties who have received computer
Xsoftware programs from you with this License Agreement will not have
Xtheir licenses terminated so long as such parties remain in full compliance.
X
X 5. If you wish to incorporate parts of GNU tar into other free
Xprograms whose distribution conditions are different, write to the Free
XSoftware Foundation at 675 Mass Ave, Cambridge, MA 02139. We have not yet
Xworked out a simple rule that can be stated here, but we will often permit
Xthis. We will be guided by the two goals of preserving the free status of
Xall derivatives of our free software and of promoting the sharing and reuse
Xof software.
X
X
XYour comments and suggestions about our licensing policies and our
Xsoftware are welcome! Please contact the Free Software Foundation, Inc.,

```
X675 Mass Ave, Cambridge, MA 02139, or call (617) 876-3296.
X
X                NO WARRANTY
X
X  BECAUSE GNU tar IS LICENSED FREE OF CHARGE, WE PROVIDE ABSOLUTELY NO
XWARRANTY, TO THE EXTENT PERMITTED BY APPLICABLE STATE LAW.  EXCEPT
XWHEN OTHERWISE STATED IN WRITING, FREE SOFTWARE FOUNDATION, INC,
XRICHARD M. STALLMAN AND/OR OTHER PARTIES PROVIDE GNU tar "AS IS" WITHOUT
XWARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT
XLIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
XA PARTICULAR PURPOSE.  THE ENTIRE RISK AS TO THE QUALITY AND
XPERFORMANCE OF GNU tar IS WITH YOU.  SHOULD GNU tar PROVE DEFECTIVE, YOU
XASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
X
X IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW WILL RICHARD M.
SHAR_EOF
echo "End of part 1"
echo "File COPYING is continued in part 2"
echo "2" > s2_seq_.tmp
exit 0


From -8300080679269729694
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,987cd08baa0a1a8b,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-19 14:42:57 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
decwrl!sun-barr!newstop!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i048: gnutar -- GNU's version of UNIX "tar"  part02/08
Keywords: shar, C
Message-ID: <33888@grapevine.uucp>
Date: 19 Jul 89 21:42:57 GMT
Sender: news@grapevine.uucp
Lines: 1793
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: 7103_300@uwovax.uwo.ca (Eric R. Smith)
Posting-number: Volume 2, Issue 48
Archive-name: gnutar/part02

#!/bin/sh
# this is part 2 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file COPYING continued
#
CurArch=2
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
```

```
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file COPYING"
sed 's/^X//' << 'SHAR_EOF' >> COPYING
XSTALLMAN, THE FREE SOFTWARE FOUNDATION, INC., AND/OR ANY OTHER PARTY
XWHO MAY MODIFY AND REDISTRIBUTE GNU tar AS PERMITTED ABOVE, BE LIABLE TO
XYOU FOR DAMAGES, INCLUDING ANY LOST PROFITS, LOST MONIES, OR OTHER
XSPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR
XINABILITY TO USE (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA
XBEING RENDERED INACCURATE OR LOSSES SUSTAINED BY THIRD PARTIES OR A
XFAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS) GNU tar, EVEN
XIF YOU HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR
XANY CLAIM BY ANY OTHER PARTY.
SHAR_EOF
echo "File COPYING is complete"
chmod 0644 COPYING || echo "restore of COPYING fails"
echo "x - extracting CREATE.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > CREATE.C &&
X/*
X
X    Copyright (C) 1988 Free Software Foundation
X
XGNU tar is distributed in the hope that it will be useful, but WITHOUT ANY
XWARRANTY.  No author or distributor accepts responsibility to anyone
Xfor the consequences of using it or for whether it serves any
Xparticular purpose or works at all, unless he says so in writing.
XRefer to the GNU tar General Public License for full details.
X
XEveryone is granted permission to copy, modify and redistribute GNU tar,
Xbut only under the conditions described in the GNU tar General Public
XLicense.  A copy of this license is supposed to have been given to you
Xalong with GNU tar so you can know your rights and responsibilities.  It
Xshould be in a file named COPYING.  Among other things, the copyright
Xnotice and this notice must be preserved on all copies.
X
XIn other words, go ahead and share GNU tar, but don't try to stop
Xanyone else from sharing it farther.  Help stamp out software hoarding!
X*/
X
X/*
X * Create a tar archive.
X *
X * Written 25 Aug 1985 by John Gilmore, ihnp4!hoptoad!gnu.
X *
X * @(#)create.c 1.36 11/6/87 - gnu
X */
X#ifdef atarist
X#include <types.h>
X#include <stat.h>
X#else
```

```
X#include <sys/types.h>
X#include <sys/stat.h>
X#endif
X#include <stdio.h>
X
X/* JF: this one is my fault */
X/* #include "utils.h" */
X
X#ifndef V7
X#include <fcntl.h>
X#endif
X
X#ifndef  MSDOS
X#include <pwd.h>
X#include <grp.h>
X#endif
X
X#ifdef BSD42
X#include <sys/dir.h>
X#else
X#if (defined(MSDOS) && !defined(atarist))
X#include <sys/dir.h>
X#else
X#ifdef USG
X#include "dirent.h"
X#define direct dirent
X#define DP_NAMELEN(x) strlen((x)->d_name)
X#else
X/*
X * FIXME: On other systems there is no standard place for the header file
X * for the portable directory access routines.  Change the #include line
X * below to bring it in from wherever it is.
X */
X#include "dirent.h"
X#define direct dirent      /* If you have the POSIX version */
X#define DP_NAMELEN(x) strlen((x)->d_name) /* ditto */
X#endif
X#endif
X#endif
X
X#ifndef DP_NAMELEN
X#define DP_NAMELEN(x)  (x)->d_namlen
X#endif
X
X#ifdef USG
X#include <sys/sysmacros.h> /* major() and minor() defined here */
X#endif
X
X/*
X * V7 doesn't have a #define for this.
X */
X#ifndef O_RDONLY
X#define  O_RDONLY  0
```

```
X#endif
X
X/*
X * Most people don't have a #define for this.
X */
X#ifndef O_BINARY
X#define O_BINARY 0
X#endif
X
X#include "tar.h"
X#include "port.h"
X
X#ifdef atarist
X#undef S_IFCHR
X#endif
X
Xextern union record *head;      /* Points to current tape header */
Xextern struct stat hstat;       /* Stat struct corresponding */
Xextern int head_standard;       /* Tape header is in ANSI format */
X
X/* JF */
Xextern struct name *gnu_list_name;
X
X/*
X * If there are no symbolic links, there is no lstat().  Use stat().
X */
X#ifndef S_IFLNK
X#define lstat stat
X#endif
X
Xextern char  *malloc();
Xextern char  *strcpy();
Xextern char  *strncpy();
Xextern void  bzero();
Xextern void  bcopy();
Xextern int   errno;
X
Xextern void print_header();
X
Xunion record *start_header();
Xvoid finish_header();
Xvoid finduname();
Xvoid findgname();
Xchar *name_next();
Xvoid to_oct();
Xvoid dump_file();
X
Xstatic nolinks;            /* Gets set if we run out of RAM */
X
Xvoid
Xcreate_archive()
X{
X    register char *p;
```

```
X    char *name_from_list();
X
X    open_archive(0);       /* Open for writing */
X
X    if(f_gnudump) {
X         char buf[MAXNAMLEN],*q,*bufp;
X
X         collect_and_sort_names();
X
X         while(p=name_from_list())
X              dump_file(p,-1);
X         /* if(!f_dironly) { */
X              blank_name_list();
X              while(p=name_from_list()) {
X                   strcpy(buf,p);
X                   if(p[strlen(p)-1]!='/')
X                        strcat(buf,"/");
X                   bufp=buf+strlen(buf);
X                   for(q=gnu_list_name->dir_contents;*q;q+=strlen(q)+1) {
X                        if(*q=='Y') {
X                             strcpy(bufp,q+1);
X                             dump_file(buf,-1);
X                        }
X                   }
X              }
X         /* } */
X
X    } else {
X         while (p = name_next(1)) {
X              dump_file(p, -1);
X         }
X    }
X
X    write_eot();
X    close_archive();
X    name_close();
X}
X
X/*
X * Dump a single file.  If it's a directory, recurse.
X * Result is 1 for success, 0 for failure.
X * Sets global "hstat" to stat() output for this file.
X */
Xvoid
Xdump_file (p, curdev)
X    char *p;           /* File name to dump */
X    int  curdev;            /* Device our parent dir was on */
X{
X    union record  *header;
X    char type;
X    extern char *save_name;     /* JF for multi-volume support */
X    extern long save_totsize;
X    extern long save_sizeleft;
```

```
X
X    extern time_t new_time;
X
X    if(f_confirm && !confirm("add",p))
X        return;
X
X    /*
X     * Use stat if following (rather than dumping) 4.2BSD's
X     * symbolic links.  Otherwise, use lstat (which, on non-4.2
X     * systems, is #define'd to stat anyway.
X     */
X    if (0 != f_follow_links? stat(p, &hstat): lstat(p, &hstat))
X    {
Xbadperror:
X        msg_perror("can't add file %s",p);
Xbadfile:
X        errors++;
X        return;
X    }
X
X    /* See if we only want new files, and check if this one is too old to
X       put in the archive. */
X    if(f_new_files && new_time>hstat.st_mtime &&
X       new_time>hstat.st_ctime && (hstat.st_mode&S_IFMT)!=S_IFDIR) {
X        if(curdev<0) {
X            msg("%s: is unchanged; not dumped",p);
X        }
X        return;
X    }
X
X    /*
X     * See if we are crossing from one file system to another,
X     * and avoid doing so if the user only wants to dump one file system.
X     */
X    if (f_local_filesys && curdev >= 0 && curdev != hstat.st_dev) {
X        msg("%s: is on a different filesystem; not dumped",p);
X        return;
X    }
X
X    /*
X     * Check for multiple links.
X     *
X     * We maintain a list of all such files that we've written so
X     * far.  Any time we see another, we check the list and
X     * avoid dumping the data again if we've done it once already.
X     */
X    if (hstat.st_nlink > 1) switch (hstat.st_mode & S_IFMT) {
X        register struct link   *lp;
X
X    case S_IFREG:           /* Regular file */
X#ifdef S_IFCTG
X    case S_IFCTG:           /* Contigous file */
X#endif
```

```
X#ifdef S_IFCHR
X    case S_IFCHR:          /* Character special file */
X#endif
X
X#ifdef S_IFBLK
X    case S_IFBLK:          /* Block     special file */
X#endif
X
X#ifdef S_IFIFO
X    case S_IFIFO:          /* Fifo      special file */
X#endif
X
X        /* First quick and dirty.  Hashing, etc later FIXME */
X        for (lp = linklist; lp; lp = lp->next) {
X             if (lp->ino == hstat.st_ino &&
X                 lp->dev == hstat.st_dev) {
X                  /* We found a link. */
X                  hstat.st_size = 0;
X                  header = start_header(p, &hstat);
X                  if (header == NULL) goto badfile;
X                  strcpy(header->header.linkname,
X                       lp->name);
X                  header->header.linkflag = LF_LINK;
X                  finish_header(header);
X        /* FIXME: Maybe remove from list after all links found? */
X                  return;       /* We dumped it */
X             }
X        }
X
X        /* Not found.  Add it to the list of possible links. */
X        lp = (struct link *) malloc( (unsigned)
X             (strlen(p) + sizeof(struct link) - NAMSIZ));
X        if (!lp) {
X             if (!nolinks) {
X                  fprintf(stderr,
X    "tar: no memory for links, they will be dumped as separate files\n");
X                  nolinks++;
X             }
X        }
X        lp->ino = hstat.st_ino;
X        lp->dev = hstat.st_dev;
X        strcpy(lp->name, p);
X        lp->next = linklist;
X        linklist = lp;
X    }
X
X    /*
X     * This is not a link to a previously dumped file, so dump it.
X     */
X    switch (hstat.st_mode & S_IFMT) {
X
X    case S_IFREG:          /* Regular file */
X#ifdef S_IFCTG
```

```
X    case S_IFCTG:          /* Contigous file */
X#endif
X    {
X         int  f;       /* File descriptor */
X         long bufsize, count;
X         register long sizeleft;
X         register union record  *start;
X
X         sizeleft = hstat.st_size;
X         /* Don't bother opening empty, world readable files. */
X         if (sizeleft > 0 || 0444 != (0444 & hstat.st_mode)) {
X              f = open(p, O_RDONLY|O_BINARY);
X              if (f < 0) goto badperror;
X         } else {
X              f = -1;
X         }
X
X         header = start_header(p, &hstat);
X         if (header == NULL) goto badfile;
X#ifdef S_IFCTG
X         /* Mark contiguous files, if we support them */
X         if (f_standard && (hstat.st_mode & S_IFMT) == S_IFCTG) {
X              header->header.linkflag = LF_CONTIG;
X         }
X#endif
X         finish_header(header);
X
X         while (sizeleft > 0) {
X              if(f_multivol) {
X                   save_name = p;
X                   save_sizeleft = sizeleft;
X                   save_totsize = hstat.st_size;
X              }
X              start = findrec();
X              bufsize = endofrecs()->charptr - start->charptr;
X              if (sizeleft < bufsize) {
X                   /* Last read -- zero out area beyond */
X                   bufsize = (int)sizeleft;
X                   count = bufsize % RECORDSIZE;
X                   if (count)
X                        bzero(start->charptr + sizeleft,
X                             (int)(RECORDSIZE - count));
X              }
X              count = read(f, start->charptr, bufsize);
X              if (count < 0) {
X                   msg_perror("read error at byte %ld, reading\
X %d bytes, in file %s",  hstat.st_size - sizeleft, bufsize,p);
X                   goto padit;
X              }
X              sizeleft -= count;
X
X              /* This is nonportable (the type of userec's arg). */
X              userec(start+(count-1)/RECORDSIZE);
```

```
X
X             if (count == bufsize) continue;
X             msg( "file %s shrunk by %d bytes, padding with zeros.\n", p,
sizeleft);
X             goto padit;        /* Short read */
X         }
X
X         if(f_multivol)
X              save_name = 0;
X
X         if (f >= 0)
X              (void)close(f);
X
X         break;
X
X         /*
X          * File shrunk or gave error, pad out tape to match
X          * the size we specified in the header.
X          */
X    padit:
X         while(sizeleft>0) {
X              save_sizeleft=sizeleft;
X              start=findrec();
X              bzero(start->charptr,RECORDSIZE);
X              userec(start);
X              sizeleft-=RECORDSIZE;
X         }
X         if(f_multivol)
X              save_name=0;
X         if(f>=0)
X              (void)close(f);
X         break;
X/*       abort(); */
X    }
X
X#ifdef S_IFLNK
X    case S_IFLNK:          /* Symbolic link */
X    {
X         int size;
X
X         hstat.st_size = 0;      /* Force 0 size on symlink */
X         header = start_header(p, &hstat);
X         if (header == NULL) goto badfile;
X         size = readlink(p, header->header.linkname, NAMSIZ);
X         if (size < 0) goto badperror;
X         if (size == NAMSIZ) {
X              msg("symbolic link %s too long\n",p);
X              break;
X         }
X         header->header.linkname[size] = '\0';
X         header->header.linkflag = LF_SYMLINK;
X         finish_header(header);      /* Nothing more to do to it */
X    }
```

```
X         break;
X#endif
X
X    case S_IFDIR:          /* Directory */
X    {
X         register DIR *dirp;
X         register struct direct *d;
X         char namebuf[NAMSIZ+2];
X         register int len;
X         int our_device = hstat.st_dev;
X
X         /* Build new prototype name */
X         strncpy(namebuf, p, sizeof (namebuf));
X         len = strlen(namebuf);
X         while (len >= 1 && '/' == namebuf[len-1])
X              len--;             /* Delete trailing slashes */
X         namebuf[len++] = '/';       /* Now add exactly one back */
X         namebuf[len] = '\0';        /* Make sure null-terminated */
X
X         /*
X          * Output directory header record with permissions
X          * FIXME, do this AFTER files, to avoid R/O dir problems?
X          * If old archive format, don't write record at all.
X          */
X         if (!f_oldarch) {
X              hstat.st_size = 0; /* Force 0 size on dir */
X              /*
X               * If people could really read standard archives,
X               * this should be:     (FIXME)
X              header = start_header(f_standard? p: namebuf, &hstat);
X               * but since they'd interpret LF_DIR records as
X               * regular files, we'd better put the / on the name.
X               */
X              header = start_header(namebuf, &hstat);
X              if (header == NULL)
X                   goto badfile; /* eg name too long */
X
X              if (f_gnudump)
X                   header->header.linkflag = LF_DUMPDIR;
X              else if (f_standard)
X                   header->header.linkflag = LF_DIR;
X
X              /* If we're gnudumping, we aren't done yet so don't close it. */
X              if(!f_gnudump)
X                   finish_header(header); /* Done with directory header */
X         }
X
X         /* Hack to remove "./" from the front of all the file names */
X         if (len == 2 && namebuf[0] == '.') {
X              len = 0;
X         }
X
X         if(f_gnudump) {
```

```
X             int sizeleft;
X             int totsize;
X             int bufsize;
X             union record *start;
X             int count;
X             char *buf,*p_buf;
X
X             buf=gnu_list_name->dir_contents; /* FOO */
X             totsize=0;
X             for(p_buf=buf;*p_buf;) {
X                  int tmp;
X
X                  tmp=strlen(p_buf)+1;
X                  totsize+=tmp;
X                  p_buf+=tmp;
X             }
X             totsize++;
X             to_oct((long)totsize,1+12,header->header.size);
X             finish_header(header);
X             p_buf=buf;
X             sizeleft=totsize;
X             while(sizeleft>0) {
X                  if(f_multivol) {
X                       save_name=p;
X                       save_sizeleft=sizeleft;
X                       save_totsize=totsize;
X                  }
X                  start=findrec();
X                  bufsize=endofrecs()->charptr - start->charptr;
X                  if(sizeleft<bufsize) {
X                       bufsize=sizeleft;
X                       count=bufsize%RECORDSIZE;
X                       if(count)
X                            bzero(start->charptr+sizeleft,RECORDSIZE-count);
X                  }
X                  bcopy(p_buf,start->charptr,bufsize);
X                  sizeleft-=bufsize;
X                  p_buf+=bufsize;
X                  userec(start+(bufsize-1)/RECORDSIZE);
X             }
X             if(f_multivol)
X                  save_name = 0;
X             break;
X        }
X
X        /* Now output all the files in the directory */
X        /* if (f_dironly)
X             break;        /* Unless the cmdline said not to */
X
X
X        errno = 0;
X        dirp = opendir(p);
X        if (!dirp) {
```

```
X             if (errno) {
X                  msg_perror ("can't open directory %s",p);
X             } else {
X                  msg("error opening directory %s",
X                      p);
X             }
X             break;
X        }
X
X        /* Should speed this up by cd-ing into the dir, FIXME */
X        while (NULL != (d=readdir(dirp))) {
X             /* Skip . and .. */
X             if(is_dot_or_dotdot(d->d_name))
X                  continue;
X
X             if (DP_NAMELEN(d) + len >= NAMSIZ) {
X                  msg("file name %s%s too long\n",
X                      namebuf, d->d_name);
X                  continue;
X             }
X             strcpy(namebuf+len, d->d_name);
X             if(f_exclude && check_exclude(namebuf))
X                  continue;
X             dump_file(namebuf, our_device);
X        }
X
X        closedir(dirp);
X    }
X        break;
X
X#ifdef S_IFCHR
X    case S_IFCHR:          /* Character special file */
X        type = LF_CHR;
X        goto easy;
X#endif
X
X#ifdef S_IFBLK
X    case S_IFBLK:          /* Block     special file */
X        type = LF_BLK;
X        goto easy;
X#endif
X
X#ifdef S_IFIFO
X    case S_IFIFO:          /* Fifo      special file */
X        type = LF_FIFO;
X#endif
X
X    easy:
X        if (!f_standard) goto unknown;
X
X        hstat.st_size = 0;     /* Force 0 size */
X        header = start_header(p, &hstat);
X        if (header == NULL) goto badfile;    /* eg name too long */
```

```
X
X         header->header.linkflag = type;
X         if (type != LF_FIFO) {
X              to_oct((long) major(hstat.st_rdev), 8,
X                   header->header.devmajor);
X              to_oct((long) minor(hstat.st_rdev), 8,
X                   header->header.devminor);
X         }
X
X         finish_header(header);
X         break;
X
X    default:
X    unknown:
X         msg("%s: Unknown file type; file ignored.\n", p);
X         break;
X    }
X}
X
X
X/*
X * Make a header block for the file  name  whose stat info is  st .
X * Return header pointer for success, NULL if the name is too long.
X */
Xunion record *
Xstart_header(name, st)
X    char *name;
X    register struct stat *st;
X{
X    register union record *header;
X
X    header = (union record *) findrec();
X    bzero(header->charptr, sizeof(*header)); /* XXX speed up */
X
X    /*
X     * Check the file name and put it in the record.
X     */
X#ifdef MSDOS
X    if(name[1]==':') {
X         static int warned_once = 0;
X         name+=2;
X         if(!warned_once++) {
X              msg("Removing drive spec from names in the archive");
X         }
X    }
X#endif
X    while ('/' == *name) {
X         static int warned_once = 0;
X
X         name++;                 /* Force relative path */
X         if (!warned_once++) {
X              msg("Removing leading / from absolute path names in the archive.");
X         }
```

```
X    }
X    strcpy(header->header.name, name);
X    if (header->header.name[NAMSIZ-1]) {
X         msg("%s: name too long\n", name);
X         return NULL;
X    }
X
X    to_oct((long) (st->st_mode & ~S_IFMT),
X                        8,  header->header.mode);
X    to_oct((long) st->st_uid,   8,  header->header.uid);
X    to_oct((long) st->st_gid,   8,  header->header.gid);
X    to_oct((long) st->st_size,  1+12, header->header.size);
X    to_oct((long) st->st_mtime, 1+12, header->header.mtime);
X    /* header->header.linkflag is left as null */
X    if(f_gnudump) {
X         to_oct((long) st->st_atime, 1+12, header->header.atime);
X         to_oct((long) st->st_ctime, 1+12, header->header.ctime);
X    }
X
X#ifndef NONAMES
X    /* Fill in new Unix Standard fields if desired. */
X    if (f_standard) {
X         header->header.linkflag = LF_NORMAL;  /* New default */
X         strcpy(header->header.magic, TMAGIC); /* Mark as Unix Std */
X         finduname(header->header.uname, st->st_uid);
X         findgname(header->header.gname, st->st_gid);
X    }
X#endif
X    return header;
X}
X
X/*
X * Finish off a filled-in header block and write it out.
X * We also print the file name and/or full info if verbose is on.
X */
Xvoid
Xfinish_header(header)
X    register union record *header;
X{
X    register int  i, sum;
X    register char *p;
X    void bcopy();
X
X    bcopy(CHKBLANKS, header->header.chksum, sizeof(header->header.chksum));
X
X    sum = 0;
X    p = header->charptr;
X    for (i = sizeof(*header); --i >= 0; ) {
X         /*
X          * We can't use unsigned char here because of old compilers,
X          * e.g. V7.
X          */
X         sum += 0xFF & *p++;
```

```
X    }
X
X    /*
X     * Fill in the checksum field.  It's formatted differently
X     * from the other fields:  it has [6] digits, a null, then a
X     * space -- rather than digits, a space, then a null.
X     * We use to_oct then write the null in over to_oct's space.
X     * The final space is already there, from checksumming, and
X     * to_oct doesn't modify it.
X     *
X     * This is a fast way to do:
X     * (void) sprintf(header->header.chksum, "%6o", sum);
X     */
X    to_oct((long) sum, 8,  header->header.chksum);
X    header->header.chksum[6] = '\0'; /* Zap the space */
X
X    userec(header);
X
X    if (f_verbose) {
X         /* These globals are parameters to print_header, sigh */
X         head = header;
X         /* hstat is already set up */
X         head_standard = f_standard;
X         print_header();
X    }
X
X    return;
X}
X
X
X/*
X * Quick and dirty octal conversion.
X * Converts long "value" into a "digs"-digit field at "where",
X * including a trailing space and room for a null.  "digs"==3 means
X * 1 digit, a space, and room for a null.
X *
X * We assume the trailing null is already there and don't fill it in.
X * This fact is used by start_header and finish_header, so don't change it!
X *
X * This should be equivalent to:
X *  (void) sprintf(where, "%*lo ", digs-2, value);
X * except that sprintf fills in the trailing null and we don't.
X */
Xvoid
Xto_oct(value, digs, where)
X    register long value;
X    register int  digs;
X    register char *where;
X{
X
X    --digs;                /* Trailing null slot is left alone */
X    where[--digs] = ' ';        /* Put in the space, though */
X
```

```
X    /* Produce the digits -- at least one */
X    do {
X         where[--digs] = '0' + (char)(value & 7); /* one octal digit */
X         value >>= 3;
X    } while (digs > 0 && value != 0);
X
X    /* Leading spaces, if necessary */
X    while (digs > 0)
X         where[--digs] = ' ';
X
X}
X
X
X/*
X * Write the EOT record(s).
X * We actually zero at least one record, through the end of the block.
X * Old tar writes garbage after two zeroed records -- and PDtar used to.
X */
Xwrite_eot()
X{
X    union record *p;
X    int bufsize;
X    void bzero();
X
X    p = findrec();
X    bufsize = endofrecs()->charptr - p->charptr;
X    bzero(p->charptr, bufsize);
X    userec(p);
X}
SHAR_EOF
chmod 0644 CREATE.C || echo "restore of CREATE.C fails"
echo "x - extracting DIFFARCH.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > DIFFARCH.C &&
X/*
X
X    Copyright (C) 1988 Free Software Foundation
X
XGNU tar is distributed in the hope that it will be useful, but WITHOUT ANY
XWARRANTY.  No author or distributor accepts responsibility to anyone
Xfor the consequences of using it or for whether it serves any
Xparticular purpose or works at all, unless he says so in writing.
XRefer to the GNU tar General Public License for full details.
X
XEveryone is granted permission to copy, modify and redistribute GNU tar,
Xbut only under the conditions described in the GNU tar General Public
XLicense.  A copy of this license is supposed to have been given to you
Xalong with GNU tar so you can know your rights and responsibilities.  It
Xshould be in a file named COPYING.  Among other things, the copyright
Xnotice and this notice must be preserved on all copies.
X
XIn other words, go ahead and share GNU tar, but don't try to stop
Xanyone else from sharing it farther.  Help stamp out software hoarding!
X*/
```

```
X
X/*
X * Diff files from a tar archive.
X *
X * Written 30 April 1987 by John Gilmore, ihnp4!hoptoad!gnu.
X *
X * @(#) diffarch.c 1.10 87/11/11 - gnu
X */
X
X#include <stdio.h>
X#include <errno.h>
X#ifdef atarist
X#include <types.h>
X#include <stat.h>
X#else
X#include <sys/types.h>
X#include <sys/stat.h>
X#endif
X
X#ifdef BSD42
X#include <sys/file.h>
X#endif
X
X#ifdef atarist
X#include <file.h>
X#endif
X
X#ifdef USG
X#include <fcntl.h>
X#endif
X
X/* Some systems don't have these #define's -- we fake it here. */
X#ifndef O_RDONLY
X#define  O_RDONLY  0
X#endif
X#ifndef  O_NDELAY
X#define  O_NDELAY  0
X#endif
X
X#ifndef S_IFLNK
X#define lstat stat
X#endif
X
Xextern int errno;          /* From libc.a */
Xextern char *valloc();          /* From libc.a */
X
X#include "tar.h"
X#include "port.h"
X#include "rmt.h"
X
Xextern union record *head;      /* Points to current tape header */
Xextern struct stat hstat;       /* Stat struct corresponding */
Xextern int head_standard;       /* Tape header is in ANSI format */
```

```
X
Xextern void print_header();
Xextern void skip_file();
X
Xextern FILE *msg_file;
X
Xint now_verifying = 0;     /* Are we verifying at the moment? */
X
Xchar    *diff_name;        /* head->header.name */
X
Xint diff_fd;      /* Descriptor of file we're diffing */
X
Xchar    *diff_buf = 0;          /* Pointer to area for reading
X                           file contents into */
X
Xchar    *diff_dir;         /* Directory contents for LF_DUMPDIR */
X
Xint different = 0;
X
X/*
X * Initialize for a diff operation
X */
Xdiff_init()
X{
X
X    /*NOSTRICT*/
X    diff_buf = (char *) valloc((unsigned)blocksize);
X    if (!diff_buf) {
X         msg("could not allocate memory for diff buffer of %d bytes\n",
X              blocking);
X         exit(EX_ARGSBAD);
X    }
X}
X
X/*
X * Diff a file against the archive.
X */
Xvoid
Xdiff_archive()
X{
X    register char *data;
X    int check, namelen;
X    int err;
X    long offset;
X    struct stat filestat;
X    char linkbuf[NAMSIZ+3];
X    int compare_chunk();
X    int compare_dir();
X    dev_t    dev;
X    ino_t    ino;
X    char *get_dir_contents();
X    long from_oct();
X    long lseek();
```

```
X
X#ifndef atarist
X    errno = EPIPE;              /* FIXME, remove perrors */
X#endif
X    saverec(&head);             /* Make sure it sticks around */
X    userec(head);          /* And go past it in the archive */
X    decode_header(head, &hstat, &head_standard, 1);    /* Snarf fields */
X
X    if(now_verifying)
X         fprintf(msg_file,"Verify ");
X    /* Print the record from 'head' and 'hstat' */
X    if (f_verbose)
X         print_header();
X
X    diff_name = head->header.name;
X    switch (head->header.linkflag) {
X
X    default:
X         msg("Unknown file type '%c' for %s, diffed as normal file\n",
X              head->header.linkflag, diff_name);
X         /* FALL THRU */
X
X    case LF_OLDNORMAL:
X    case LF_NORMAL:
X    case LF_CONTIG:
X         /*
X          * Appears to be a file.
X          * See if it's really a directory.
X          */
X         namelen = strlen(diff_name)-1;
X         if (diff_name[namelen] == '/')
X              goto really_dir;
X
X
X         if(do_stat(&filestat))
X              goto quit;
X
X         if ((filestat.st_mode & S_IFMT) != S_IFREG) {
X              fprintf(msg_file, "%s: not a regular file\n",
X                   diff_name);
X              skip_file((long)hstat.st_size);
X              different++;
X              goto quit;
X         }
X
X         filestat.st_mode &= ~S_IFMT;
X         if (filestat.st_mode != hstat.st_mode)
X              sigh("mode");
X         if (filestat.st_uid  != hstat.st_uid)
X              sigh("uid");
X         if (filestat.st_gid  != hstat.st_gid)
X              sigh("gid");
X         if (filestat.st_mtime != hstat.st_mtime)
```

```
X             sigh("mod time");
X         if (filestat.st_size != hstat.st_size) {
X             sigh("size");
X             skip_file((long)hstat.st_size);
X             goto quit;
X         }
X
X         diff_fd = open(diff_name, O_NDELAY|O_RDONLY);
X
X         if (diff_fd < 0) {
X             msg_perror("cannot open %s",diff_name);
X             skip_file((long)hstat.st_size);
X             different++;
X             goto quit;
X         }
X
X         wantbytes((long)(hstat.st_size),compare_chunk);
X
X         check = close(diff_fd);
X         if (check < 0) {
X             msg_perror("Error while closing %s",diff_name);
X         }
X
X    quit:
X         break;
X
X    case LF_LINK:
X         if(do_stat(&filestat))
X             break;
X         dev = filestat.st_dev;
X         ino = filestat.st_ino;
X         err = stat(head->header.linkname, &filestat);
X         if (err < 0) {
X             if (errno==ENOENT) {
X                 fprintf(msg_file, "%s: does not exist\n",diff_name);
X             } else {
X                 msg_perror("cannot stat file %s",diff_name);
X             }
X             different++;
X             break;
X         }
X         if(filestat.st_dev!=dev || filestat.st_ino!=ino) {
X             fprintf(msg_file, "%s not linked to %s\n",diff_name,head-
>header.linkname);
X             break;
X         }
X         break;
X
X#ifdef S_IFLNK
X    case LF_SYMLINK:
X         check = readlink(diff_name, linkbuf,
X                   (sizeof linkbuf)-1);
X
```

```
X         if (check < 0) {
X              if (errno == ENOENT) {
X                   fprintf(msg_file,
X                        "%s: no such file or directory\n",
X                        diff_name);
X              } else {
X                   msg_perror("cannot read link %s",diff_name);
X              }
X              different++;
X              break;
X         }
X
X         linkbuf[check] = '\0'; /* Null-terminate it */
X         if (strncmp(head->header.linkname, linkbuf, check) != 0) {
X              fprintf(msg_file, "%s: symlink differs\n",
X                   head->header.linkname);
X              different++;
X         }
X         break;
X#endif
X
X    case LF_CHR:
X         hstat.st_mode |= S_IFCHR;
X         goto check_node;
X
X#ifdef S_IFBLK
X    /* If local system doesn't support block devices, use default case */
X    case LF_BLK:
X         hstat.st_mode |= S_IFBLK;
X         goto check_node;
X#endif
X
X#ifdef S_IFIFO
X    /* If local system doesn't support FIFOs, use default case */
X    case LF_FIFO:
X         hstat.st_mode |= S_IFIFO;
X         hstat.st_rdev = 0;     /* FIXME, do we need this? */
X         goto check_node;
X#endif
X
X    check_node:
X         /* FIXME, deal with umask */
X         if(do_stat(&filestat))
X              break;
X         if(hstat.st_rdev != filestat.st_rdev) {
X              fprintf(msg_file, "%s: device numbers changed\n", diff_name);
X              different++;
X              break;
X         }
X         if(hstat.st_mode != filestat.st_mode) {
X              fprintf(msg_file, "%s: mode or device-type changed\n", diff_name);
X              different++;
X              break;
```

```
X         }
X         break;
X
X    case LF_DUMPDIR:
X         data=diff_dir=get_dir_contents(diff_name,0);
X         wantbytes((long)(hstat.st_size),compare_dir);
X         free(data);
X         /* FALL THROUGH */
X
X    case LF_DIR:
X         /* Check for trailing / */
X         namelen = strlen(diff_name)-1;
X    really_dir:
X         while (namelen && diff_name[namelen] == '/')
X              diff_name[namelen--] = '\0';     /* Zap / */
X
X         if(do_stat(&filestat))
X              break;
X         if((filestat.st_mode&S_IFMT)!=S_IFDIR) {
X              fprintf(msg_file, "%s is no longer a directory\n",diff_name);
X              different++;
X              break;
X         }
X         if((filestat.st_mode&~S_IFMT) != hstat.st_mode)
X              sigh("mode");
X         break;
X
X    case LF_VOLHDR:
X         break;
X
X    case LF_MULTIVOL:
X         namelen = strlen(diff_name)-1;
X         if (diff_name[namelen] == '/')
X              goto really_dir;
X
X         if(do_stat(&filestat))
X              break;
X
X         if ((filestat.st_mode & S_IFMT) != S_IFREG) {
X              fprintf(msg_file, "%s: not a regular file\n",
X                   diff_name);
X              skip_file((long)hstat.st_size);
X              different++;
X              break;
X         }
X
X         filestat.st_mode &= ~S_IFMT;
X         offset = from_oct(1+12, head->header.offset);
X         if (filestat.st_size != hstat.st_size + offset) {
X              sigh("size");
X              skip_file((long)hstat.st_size);
X              different++;
X              break;
```

```
X        }
X
X        diff_fd = open(diff_name, O_NDELAY|O_RDONLY);
X
X        if (diff_fd < 0) {
X             msg_perror("cannot open file %s",diff_name);
X             skip_file((long)hstat.st_size);
X             different++;
X             break;
X        }
X        err = lseek(diff_fd, offset, 0);
X        if(err!=offset) {
X             msg_perror("cannot seek to %ld in file %s",offset,diff_name);
X             different++;
X             break;
X        }
X
X        wantbytes((long)(hstat.st_size),compare_chunk);
X
X        check = close(diff_fd);
X        if (check < 0) {
X             msg_perror("Error while closing %s",diff_name);
X        }
X        break;
X
X    }
X
X    /* We don't need to save it any longer. */
X    saverec((union record **) 0);    /* Unsave it */
X}
X
Xint
Xcompare_chunk(bytes,buffer)
Xint bytes;
Xchar *buffer;
X{
X    int err;
X
X    err=read(diff_fd,diff_buf,bytes);
X    if(err!=bytes) {
X        if(err<0) {
X             msg_perror("can't read %s",diff_name);
X        } else {
X             fprintf(msg_file,"%s: could only read %d of %d bytes\n",err,bytes);
X        }
X        different++;
X        return -1;
X    }
X    if(bcmp(buffer,diff_buf,bytes)) {
X        fprintf(msg_file, "%s: data differs\n",diff_name);
X        different++;
X        return -1;
X    }
```

```
X    return 0;
X}
X
Xint
Xcompare_dir(bytes,buffer)
Xint bytes;
Xchar *buffer;
X{
X    if(bcmp(buffer,diff_dir,bytes)) {
X         fprintf(msg_file, "%s: data differs\n",diff_name);
X         different++;
X         return -1;
X    }
X    diff_dir+=bytes;
X    return 0;
X}
X
X/*
X * Sigh about something that differs.
X */
Xsigh(what)
X    char *what;
X{
X
X    fprintf(msg_file, "%s: %s differs\n",
X         diff_name, what);
X}
X
Xverify_volume()
X{
X    int status;
X#ifdef MTIOCTOP
X    struct mtop t;
X    int er;
X#endif
X
X    if(!diff_buf)
X         diff_init();
X#ifdef MTIOCTOP
X    t.mt_op = MTBSF;
X    t.mt_count = 1;
X    if((er=rmtioctl(archive,MTIOCTOP,&t))<0) {
X         if(errno!=EIO || (er=rmtioctl(archive,MTIOCTOP,&t))<0) {
X#endif
X              if(rmtlseek(archive,0L,0)!=0) {
X                   /* Lseek failed.  Try a different method */
X                   msg_perror("Couldn't rewind archive file for verify");
X                   return;
X              }
X#ifdef MTIOCTOP
X         }
X    }
X#endif
```

```
X    ar_reading=1;
X    now_verifying = 1;
X    fl_read();
X    for(;;) {
X         status = read_header();
X         if(status==0) {
X              unsigned n;
X
X              n=0;
X              do {
X                   n++;
X                   status=read_header();
X              } while(status==0);
X              msg("VERIFY FAILURE: %d invalid header%s detected!",n,n==1?"":"s");
X         }
X         if(status==2 || status==EOF)
X              break;
X         diff_archive();
X    }
X    ar_reading=0;
X    now_verifying = 0;
X}
X
Xint do_stat(statp)
Xstruct stat *statp;
X{
X    int err;
X
X    err = f_follow_links ? stat(diff_name, statp) : lstat(diff_name, statp);
X    if (err < 0) {
X         if (errno==ENOENT) {
X              fprintf(msg_file, "%s: does not exist\n",diff_name);
X         } else
X              msg_perror("can't stat file %s",diff_name);
X         skip_file((long)hstat.st_size);
X         different++;
X         return 1;
X    } else
X         return 0;
X}
SHAR_EOF
chmod 0644 DIFFARCH.C || echo "restore of DIFFARCH.C fails"
echo "x - extracting EXTRACT.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > EXTRACT.C &&
X/*
X
X    Copyright (C) 1988 Free Software Foundation
X
XGNU tar is distributed in the hope that it will be useful, but WITHOUT ANY
XWARRANTY.  No author or distributor accepts responsibility to anyone
Xfor the consequences of using it or for whether it serves any
Xparticular purpose or works at all, unless he says so in writing.
XRefer to the GNU tar General Public License for full details.
```

```
X
XEveryone is granted permission to copy, modify and redistribute GNU tar,
Xbut only under the conditions described in the GNU tar General Public
XLicense.  A copy of this license is supposed to have been given to you
Xalong with GNU tar so you can know your rights and responsibilities.  It
Xshould be in a file named COPYING.  Among other things, the copyright
Xnotice and this notice must be preserved on all copies.
X
XIn other words, go ahead and share GNU tar, but don't try to stop
Xanyone else from sharing it farther.  Help stamp out software hoarding!
X*/
X
X/*
X * Extract files from a tar archive.
X *
X * Written 19 Nov 1985 by John Gilmore, ihnp4!hoptoad!gnu.
X *
X * @(#) extract.c 1.32 87/11/11 - gnu
X */
X
X#include <stdio.h>
X#include <errno.h>
X#ifdef atarist
X#include <types.h>
X#include <stat.h>
X#else
X#include <sys/types.h>
X#include <sys/stat.h>
X#endif
X
X#ifdef BSD42
X#include <sys/file.h>
X#endif
X
X#ifdef USG
X#include <fcntl.h>
X#endif
X
X#ifdef  MSDOS
X#include <fcntl.h>
X#endif  /* MSDOS */
X
X/*
X * Some people don't have a #define for these.
X */
X#ifndef O_BINARY
X#define O_BINARY 0
X#endif
X#ifndef O_NDELAY
X#define O_NDELAY 0
X#endif
X
X#ifdef NO_OPEN3
```

```
X/* We need the #define's even though we don't use them. */
X#include "open3.h"
X#endif
X
X#ifdef EMUL_OPEN3
X/* Simulated 3-argument open for systems that don't have it */
X#include "open3.h"
X#endif
X
Xextern int errno;          /* From libc.a */
Xextern time_t time();           /* From libc.a */
Xextern char *index();           /* From libc.a or port.c */
X
X#include "tar.h"
X#include "port.h"
X
Xextern FILE *msg_file;
X
Xextern union record *head;      /* Points to current tape header */
Xextern struct stat hstat;       /* Stat struct corresponding */
Xextern int head_standard;       /* Tape header is in ANSI format */
X
Xextern char *save_name;
Xextern long save_totsize;
Xextern long save_sizeleft;
X
Xextern void print_header();
Xextern void skip_file();
Xextern void pr_mkdir();
X
Xint make_dirs();           /* Makes required directories */
X
Xstatic time_t now = 0;          /* Current time */
Xstatic we_are_root = 0;             /* True if our effective uid == 0 */
Xstatic int notumask = ~0;       /* Masks out bits user doesn't want */
X
X/*
X * Set up to extract files.
X */
Xextr_init()
X{
X    int ourmask;
X
X    now = time((time_t *)0);
X    if (geteuid() == 0)
X        we_are_root = 1;
X
X    /*
X     * We need to know our umask.  But if f_use_protection is set,
X     * leave our kernel umask at 0, and our "notumask" at ~0.
X     */
X    ourmask = umask(0);         /* Read it */
X    if (!f_use_protection) {
```

```
X         (void) umask (ourmask); /* Set it back how it was */
X         notumask = ~ourmask;    /* Make umask override permissions */
X    }
X}
X
X
X/*
X * Extract a file from the archive.
X */
Xvoid
Xextract_archive()
X{
X    register char *data;
X    int fd, check, namelen, written, openflag;
X    long size;
X    time_t acc_upd_times[2];
X    register int skipcrud;
X
X    saverec(&head);             /* Make sure it sticks around */
X    userec(head);          /* And go past it in the archive */
X    decode_header(head, &hstat, &head_standard, 1);    /* Snarf fields */
X
X    if(f_confirm && !confirm("extract",head->header.name)) {
X         skip_file((long)hstat.st_size);
X         saverec((union record **)0);
X         return;
X    }
X
X    /* Print the record from 'head' and 'hstat' */
X    if (f_verbose)
X         print_header();
X
X    /*
X     * Check for fully specified pathnames and other atrocities.
X     *
X     * Note, we can't just make a pointer to the new file name,
X     * since saverec() might move the header and adjust "head".
X     * We have to start from "head" every time we want to touch
X     * the header record.
X     */
X    skipcrud = 0;
X    while ('/' == head->header.name[skipcrud]) {
X         static int warned_once = 0;
X
X         skipcrud++;   /* Force relative path */
X         if (!warned_once++) {
X              msg("Removing leading / from absolute path names in the archive.");
X         }
X    }
X
X    switch (head->header.linkflag) {
X
X    default:
```

```
X         msg("Unknown file type '%c' for %s, extracted as normal file",
X             head->header.linkflag, skipcrud+head->header.name);
X         /* FALL THRU */
X
X    case LF_OLDNORMAL:
X    case LF_NORMAL:
X    case LF_CONTIG:
X         /*
X          * Appears to be a file.
X          * See if it's really a directory.
X          */
X         namelen = strlen(skipcrud+head->header.name)-1;
X         if (head->header.name[skipcrud+namelen] == '/')
X             goto really_dir;
X
X         /* FIXME, deal with protection issues */
X    again_file:
X         openflag = f_keep?
X             O_BINARY|O_NDELAY|O_WRONLY|O_APPEND|O_CREAT|O_EXCL:
X             O_BINARY|O_NDELAY|O_WRONLY|O_APPEND|O_CREAT|O_TRUNC;
X         if(f_exstdout) {
X             fd = 1;
X             goto extract_file;
X         }
X#ifdef O_CTG
X         /*
X          * Contiguous files (on the Masscomp) have to specify
X          * the size in the open call that creates them.
X          */
X         if (head->header.lnkflag == LF_CONTIG)
X             fd = open(skipcrud+head->header.name, openflag | O_CTG,
X                  hstat.st_mode, hstat.st_size);
X         else
X#endif
X         {
X#ifdef NO_OPEN3
X             /*
X              * On raw V7 we won't let them specify -k (f_keep), but
X              * we just bull ahead and create the files.
X              */
X             fd = creat(skipcrud+head->header.name,
X                  hstat.st_mode);
X#else
X             /*
X              * With 3-arg open(), we can do this up right.
X              */
X             fd = open(skipcrud+head->header.name, openflag,
X                  hstat.st_mode);
X#endif
X         }
X
X         if (fd < 0) {
X             if (make_dirs(skipcrud+head->header.name))
```

```
X                 goto again_file;
X             msg_perror("Could not create file %s",skipcrud+head->header.name);
X             skip_file((long)hstat.st_size);
X             goto quit;
X         }
X
X    extract_file:
X         for (size = hstat.st_size;
X              size > 0;
X              size -= written) {
X
X             if(f_multivol) {
X                  save_name=head->header.name;
X                  save_totsize=hstat.st_size;
X                  save_sizeleft=size;
X             }
X
X             /*
X              * Locate data, determine max length
X              * writeable, write it, record that
X              * we have used the data, then check
X              * if the write worked.
X              */
X             data = findrec()->charptr;
X             if (data == NULL) {    /* Check it... */
X                  msg("Unexpected EOF on archive file");
X                  break;
X             }
X             written = endofrecs()->charptr - data;
X             if (written > size)
X                  written = size;
X             errno = 0;
X             check = write(fd, data, written);
X             /*
X              * The following is in violation of strict
X              * typing, since the arg to userec
X              * should be a struct rec *.  FIXME.
X              */
X             userec((union record *)(data + written - 1));
X             if (check == written) continue;
X             /*
X              * Error in writing to file.
X              * Print it, skip to next file in archive.
X              */
X             if(check<0)
X                  msg_perror("couldn't write to file %s",skipcrud+head-
>header.name);
X             else
X                  msg("could only write %d of %d bytes to file
%s",written,check,skipcrud+head->header.name);
X             skip_file((long)(size - written));
X             break;    /* Still do the close, mod time, chmod, etc */
X         }
```

```
X
X        if(f_multivol)
X             save_name = 0;
X
X             /* If writing to stdout, don't try to do anything
X                to the filename; it doesn't exist, or we don't
X                want to touch it anyway */
X        if(f_exstdout)
X             break;
X
X        check = close(fd);
X        if (check < 0) {
X             msg_perror("Error while closing %s",skipcrud+head->header.name);
X        }
X
X   set_filestat:
X#ifdef atarist
X    /* GEMDOS doesn't allow changing attributes of directories */
X        if (hstat.st_mode & S_IFDIR)
X             break;
X#endif
X        /*
X         * Set the modified time of the file.
X         *
X         * Note that we set the accessed time to "now", which
X         * is really "the time we started extracting files".
X         * unless f_gnudump is used, in which case .st_atime is used
X         */
X        if (!f_modified) {
X             /* fixme if f_gnudump should set ctime too, but how? */
X             if(f_gnudump) acc_upd_times[0]=hstat.st_atime;
X             else acc_upd_times[0] = now;              /* Accessed now */
X             acc_upd_times[1] = hstat.st_mtime; /* Mod'd */
X             if (utime(skipcrud+head->header.name,
X                 acc_upd_times) < 0) {
X                  msg_perror("couldn't change access and modification times of
%s",skipcrud+head->header.name);
X             }
X        }
X
X        /*
X         * If we are root, set the owner and group of the extracted
X         * file.  This does what is wanted both on real Unix and on
X         * System V.  If we are running as a user, we extract as that
X         * user; if running as root, we extract as the original owner.
X         */
X        if (we_are_root) {
X             if (chown(skipcrud+head->header.name, hstat.st_uid,
X                    hstat.st_gid) < 0) {
X                  msg_perror("cannot chown file %s to uid %d gid
%d",skipcrud+head->header.name,hstat.st_uid,hstat.st_gid);
X             }
X        }
```

```
X
X        /*
X         * If '-k' is not set, open() or creat() could have saved
X         * the permission bits from a previously created file,
X         * ignoring the ones we specified.
X         * Even if -k is set, if the file has abnormal
X         * mode bits, we must chmod since writing or chown() has
X         * probably reset them.
X         *
X         * If -k is set, we know *we* created this file, so the mode
X         * bits were set by our open().   If the file is "normal", we
X         * skip the chmod.  This works because we did umask(0) if -p
X         * is set, so umask will have left the specified mode alone.
X         */
X        if ((!f_keep)
X            || (hstat.st_mode & (S_ISUID|S_ISGID|S_ISVTX))) {
X             if (chmod(skipcrud+head->header.name,
X                    notumask & (int)hstat.st_mode) < 0) {
X                 msg_perror("cannot change mode of file %s to
%ld",skipcrud+head->header.name,notumask & (int)hstat.st_mode);
X             }
X        }
X
X    quit:
X        break;
X
X    case LF_LINK:
X    again_link:
X        check = link (head->header.linkname,
X                   skipcrud+head->header.name);
X        if (check == 0)
X            break;
X        if (make_dirs(skipcrud+head->header.name))
X            goto again_link;
X        if(f_gnudump && errno==EEXIST)
X            break;
X        msg_perror("Could not link %s to %s",
X            skipcrud+head->header.name,head->header.linkname);
X        break;
X
X#ifdef S_IFLNK
X    case LF_SYMLINK:
X    again_symlink:
X        check = symlink(head->header.linkname,
X                     skipcrud+head->header.name);
X        /* FIXME, don't worry uid, gid, etc... */
X        if (check == 0)
X            break;
X        if (make_dirs(skipcrud+head->header.name))
X            goto again_symlink;
X        msg_perror("Could not create symlink to %s",head->header.linkname);
X        break;
X#endif
```

```
X
X#ifdef S_IFCHR
X    case LF_CHR:
X         hstat.st_mode |= S_IFCHR;
X         goto make_node;
X#endif
X
X#ifdef S_IFBLK
X    case LF_BLK:
X         hstat.st_mode |= S_IFBLK;
X         goto make_node;
X#endif
X
X#ifdef S_IFIFO
X    /* If local system doesn't support FIFOs, use default case */
X    case LF_FIFO:
X         hstat.st_mode |= S_IFIFO;
X         hstat.st_rdev = 0;      /* FIXME, do we need this? */
X         goto make_node;
X#endif
X
X    make_node:
X         check = mknod(skipcrud+head->header.name,
X                   (int) hstat.st_mode, (int) hstat.st_rdev);
X         if (check != 0) {
X              if (make_dirs(skipcrud+head->header.name))
X                   goto make_node;
X              msg_perror("Could not make %s",skipcrud+head->header.name);
X              break;
X         };
X         goto set_filestat;
X
X    case LF_DIR:
X    case LF_DUMPDIR:
X         namelen = strlen(skipcrud+head->header.name)-1;
X    really_dir:
X         /* Check for trailing /, and zap as many as we find. */
X         while (namelen && head->header.name[skipcrud+namelen] == '/')
X              head->header.name[skipcrud+namelen--] = '\0';
X         if(f_gnudump) {        /* Read the entry and delete files
X                             that aren't listed in the archive */
X              gnu_restore(skipcrud+head->header.name);
X
X         } else if(head->header.linkflag==LF_DUMPDIR)
X              skip_file((long)(hstat.st_size));
X
X
X    again_dir:
X         check = mkdir(skipcrud+head->header.name,
X                   0300 | (int)hstat.st_mode);
X         if (check != 0) {
X              if (make_dirs(skipcrud+head->header.name))
X                   goto again_dir;
```

```
X             /* If we're trying to create '.', let it be. */
X             if (head->header.name[skipcrud+namelen] == '.' &&
X                 (namelen==0 ||
X                  head->header.name[skipcrud+namelen-1]=='/'))
X                  goto check_perms;
X             if(f_gnudump && errno==EEXIST)
X                  break;
X             msg_perror("Could not make directory %s",skipcrud+head-
>header.name);
X             break;
X         }
X
X    check_perms:
X         if (0300 != (0300 & (int) hstat.st_mode)) {
X             hstat.st_mode |= 0300;
X             msg("Added write and execute permission to directory %s",
X               skipcrud+head->header.name);
X         }
X#ifdef atarist
X         hstat.st_mode |= S_IFDIR;
X#endif
X         goto set_filestat;
X         /* FIXME, Remember timestamps for after files created? */
X         /* FIXME, change mode after files created (if was R/O dir) */
X    case LF_VOLHDR:
X         if(f_verbose) {
X             printf("Reading %s\n",head->header.name);
X         }
X    case LF_MULTIVOL:
X         msg("Can't extract '%s'--file is continued from another volume\n",head-
>header.name);
X         skip_file((long)hstat.st_size);
X         break;
X
X    }
X
X    /* We don't need to save it any longer. */
X    saverec((union record **) 0);    /* Unsave it */
X}
X
X/*
X * After a file/link/symlink/dir creation has failed, see if
X * it's because some required directory was not present, and if
X * so, create all required dirs.
X */
Xint
Xmake_dirs(pathname)
X    char *pathname;
X{
X    char *p;           /* Points into path */
X    int madeone = 0;       /* Did we do anything yet? */
X    int save_errno = errno;     /* Remember caller's errno */
X    int check;
```

```
X
X#ifdef atarist
X    if (errno != EPATH)
X#else
X    if (errno != ENOENT)
X#endif
X        return 0;     /* Not our problem */
X
X    for (p = index(pathname, '/'); p != NULL; p = index(p+1, '/')) {
X        /* Avoid mkdir of empty string, if leading or double '/' */
X        if (p == pathname || p[-1] == '/')
X            continue;
X        /* Avoid mkdir where last part of path is '.' */
X        if (p[-1] == '.' && (p == pathname+1 || p[-2] == '/'))
X            continue;
X        *p = 0;                 /* Truncate the path there */
X        check = mkdir (pathname, 0777);  /* Try to create it as a dir */
X        if (check == 0) {
X            /* Fix ownership */
X            if (we_are_root) {
X                if (chown(pathname, hstat.st_uid,
X                       hstat.st_gid) < 0) {
X                    msg_perror("cannot change owner of %s to uid %d gid
%d",pathname,hstat.st_uid,hstat.st_gid);
X                }
X            }
X            pr_mkdir(pathname, p-pathname, notumask&0777);
X            madeone++;         /* Remember if we made one */
X            *p = '/';
X            continue;
X        }
X        *p = '/';
X        if (errno == EEXIST)        /* Directory already exists */
X            continue;
X        /*
SHAR_EOF
echo "End of part 2"
echo "File EXTRACT.C is continued in part 3"
echo "3" > s2_seq_.tmp
exit 0


From -7979650564485050674
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,b470fd901efceb1e,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-19 14:43:06 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!iuvax!rutgers!apple!sun-barr!
newstop!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i049: gnutar -- GNU's version of UNIX "tar"  part03/08
Keywords: shar, C
```

```
Message-ID: <33889@grapevine.uucp>
Date: 19 Jul 89 21:43:06 GMT
Sender: news@grapevine.uucp
Lines: 1871
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: 7103_300@uwovax.uwo.ca (Eric R. Smith)
Posting-number: Volume 2, Issue 49
Archive-name: gnutar/part03

#!/bin/sh
# this is part 3 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file EXTRACT.C continued
#
CurArch=3
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file EXTRACT.C"
sed 's/^X//' << 'SHAR_EOF' >> EXTRACT.C
X         * Some other error in the mkdir.  We return to the caller.
X         */
X        break;
X    }
X
X    errno = save_errno;         /* Restore caller's errno */
X    return madeone;             /* Tell them to retry if we made one */
X}
SHAR_EOF
echo "File EXTRACT.C is complete"
chmod 0644 EXTRACT.C || echo "restore of EXTRACT.C fails"
echo "x - extracting GETDATE.Y (Text)"
sed 's/^X//' << 'SHAR_EOF' > GETDATE.Y &&
X%token ID MONTH DAY MERIDIAN NUMBER UNIT MUNIT SUNIT ZONE DAYZONE AGO
X%{
X    /*   Originally from: Steven M. Bellovin (unc!smb)  */
X    /*   Dept. of Computer Science            */
X    /*   University of North Carolina at Chapel Hill    */
X    /*   @(#)getdate.y 2.17 11/30/87          */
X
X/* #include "defs.h" JF not used any more */
X#include <sys/types.h>
X#ifdef USG
Xstruct timeb
X{
X    time_t   time;
```

```
X    unsigned short millitm;
X    short    timezone;
X    short    dstflag;
X};
X#else
X#include <sys/timeb.h>
X#endif
X#include <ctype.h>
X
X#if defined(BSD4_2) || defined (BSD4_1C)
X#include <sys/time.h>
X#else /* sane */
X#include <time.h>
X#endif /* sane */
X
X#define  NULL 0
X#define daysec (24L*60L*60L)
X    static int timeflag, zoneflag, dateflag, dayflag, relflag;
X    static time_t relsec, relmonth;
X    static int hh, mm, ss, merid, day_light;
X    static int dayord, dayreq;
X    static int month, day, year;
X    static int ourzone;
X#define AM 1
X#define PM 2
X#define DAYLIGHT 1
X#define STANDARD 2
X#define MAYBE    3
X
Xstatic time_t timeconv();
Xstatic time_t daylcorr();
Xstatic lookup();
X
X%}
X
X%%
Xtimedate:         /* empty */
X    | timedate item
X    ;
X
Xitem:   tspec
X        {timeflag++;}
X    | zone
X        {zoneflag++;}
X    | dtspec
X        {dateflag++;}
X    | dyspec
X        {dayflag++;}
X    | rspec
X        {relflag++;}
X    | nspec;
X
Xnspec:  NUMBER
```

```
X        {if (timeflag && dateflag && !relflag) year = $1;
X        else {timeflag++;hh = $1/100;mm = $1%100;ss = 0;merid = 24;}};
X
Xtspec:  NUMBER MERIDIAN
X        {hh = $1; mm = 0; ss = 0; merid = $2;}
X    | NUMBER ':' NUMBER
X        {hh = $1; mm = $3; merid = 24;}
X    | NUMBER ':' NUMBER MERIDIAN
X        {hh = $1; mm = $3; merid = $4;}
X    | NUMBER ':' NUMBER NUMBER
X        {hh = $1; mm = $3; merid = 24;
X        day_light = STANDARD; ourzone = -($4%100 + 60*($4/100));}
X    | NUMBER ':' NUMBER ':' NUMBER
X        {hh = $1; mm = $3; ss = $5; merid = 24;}
X    | NUMBER ':' NUMBER ':' NUMBER MERIDIAN
X        {hh = $1; mm = $3; ss = $5; merid = $6;}
X    | NUMBER ':' NUMBER ':' NUMBER NUMBER
X        {hh = $1; mm = $3; ss = $5; merid = 24;
X        day_light = STANDARD; ourzone = -($6%100 + 60*($6/100));};
X
Xzone:   ZONE
X        {ourzone = $1; day_light = STANDARD;}
X    | DAYZONE
X        {ourzone = $1; day_light = DAYLIGHT;};
X
Xdyspec: DAY
X        {dayord = 1; dayreq = $1;}
X    | DAY ','
X        {dayord = 1; dayreq = $1;}
X    | NUMBER DAY
X        {dayord = $1; dayreq = $2;};
X
Xdtspec: NUMBER '/' NUMBER
X        {month = $1; day = $3;}
X    | NUMBER '/' NUMBER '/' NUMBER
X        {month = $1; day = $3; year = $5;}
X    | MONTH NUMBER
X        {month = $1; day = $2;}
X    | MONTH NUMBER ',' NUMBER
X        {month = $1; day = $2; year = $4;}
X    | NUMBER MONTH
X        {month = $2; day = $1;}
X    | NUMBER MONTH NUMBER
X        {month = $2; day = $1; year = $3;};
X
X
Xrspec:  NUMBER UNIT
X        {relsec +=  60L * $1 * $2;}
X    | NUMBER MUNIT
X        {relmonth += $1 * $2;}
X    | NUMBER SUNIT
X        {relsec += $1;}
X    | UNIT
```

```
X        {relsec +=  60L * $1;}
X    | MUNIT
X        {relmonth += $1;}
X    | SUNIT
X        {relsec++;}
X    | rspec AGO
X        {relsec = -relsec; relmonth = -relmonth;};
X%%
X
Xstatic int mdays[12] =
X    {31, 0, 31,  30, 31, 30,  31, 31, 30,  31, 30, 31};
X#define epoch 1970
X
Xextern struct tm *localtime();
X
Xstatic time_t
Xdateconv(mm, dd, yy, h, m, s, mer, zone, dayflag)
Xint mm, dd, yy, h, m, s, mer, zone, dayflag;
X{
X    time_t tod, jdate;
X    register int i;
X
X    if (yy < 0) yy = -yy;
X    if (yy < 100) yy += 1900;
X    mdays[1] = 28 + (yy%4 == 0 && (yy%100 != 0 || yy%400 == 0));
X    if (yy < epoch || yy > 1999 || mm < 1 || mm > 12 ||
X        dd < 1 || dd > mdays[--mm]) return (-1);
X    jdate = dd-1;
X        for (i=0; i<mm; i++) jdate += mdays[i];
X    for (i = epoch; i < yy; i++) jdate += 365 + (i%4 == 0);
X    jdate *= daysec;
X    jdate += zone * 60L;
X    if ((tod = timeconv(h, m, s, mer)) < 0) return (-1);
X    jdate += tod;
X    if (dayflag==DAYLIGHT || (dayflag==MAYBE&&localtime(&jdate)->tm_isdst))
X        jdate += -1*60*60;
X    return (jdate);
X}
X
Xstatic time_t
Xdayconv(ord, day, now)
Xint ord, day; time_t now;
X{
X    register struct tm *loctime;
X    time_t tod;
X
X    tod = now;
X    loctime = localtime(&tod);
X    tod += daysec * ((day - loctime->tm_wday + 7) % 7);
X    tod += 7*daysec*(ord<=0?ord:ord-1);
X    return daylcorr(tod, now);
X}
X
```

```
Xstatic time_t
Xtimeconv(hh, mm, ss, mer)
Xregister int hh, mm, ss, mer;
X{
X    if (mm < 0 || mm > 59 || ss < 0 || ss > 59) return (-1);
X    switch (mer) {
X         case AM: if (hh < 1 || hh > 12) return(-1);
X              return (60L * ((hh%12)*60L + mm)+ss);
X         case PM: if (hh < 1 || hh > 12) return(-1);
X              return (60L * ((hh%12 +12)*60L + mm)+ss);
X         case 24: if (hh < 0 || hh > 23) return (-1);
X              return (60L * (hh*60L + mm)+ss);
X         default: return (-1);
X    }
X}
X
Xstatic time_t
Xmonthadd(sdate, relmonth)
Xtime_t sdate, relmonth;
X{
X    struct tm *ltime;
X    time_t dateconv();
X    int mm, yy;
X
X    if (relmonth == 0) return 0;
X    ltime = localtime(&sdate);
X    mm = 12*ltime->tm_year + ltime->tm_mon + relmonth;
X    yy = mm/12;
X    mm = mm%12 + 1;
X    return daylcorr(dateconv(mm, ltime->tm_mday, yy, ltime->tm_hour,
X         ltime->tm_min, ltime->tm_sec, 24, ourzone, MAYBE), sdate);
X}
X
Xstatic time_t
Xdaylcorr(future, now)
Xtime_t future, now;
X{
X    int fdayl, nowdayl;
X
X    nowdayl = (localtime(&now)->tm_hour+1) % 24;
X    fdayl = (localtime(&future)->tm_hour+1) % 24;
X    return (future-now) + 60L*60L*(nowdayl-fdayl);
X}
X
Xstatic char *lptr;
X
Xstatic yylex()
X{
X    extern int yylval;
X    int sign;
X    register char c;
X    register char *p;
X    char idbuf[20];
```

```
X    int pcnt;
X
X    for (;;) {
X         while (isspace(*lptr))
X              lptr++;
X
X         if (isdigit(c = *lptr) || c == '-' || c == '+') {
X              if (c== '-' || c == '+') {
X                   if (c=='-') sign = -1;
X                   else sign = 1;
X                   if (!isdigit(*++lptr)) {
X                        /* yylval = sign; return (NUMBER); */
X                        return yylex();    /* skip the '-' sign */
X                   }
X              } else sign = 1;
X              yylval = 0;
X              while (isdigit(c = *lptr++))
X                   yylval = 10*yylval + c - '0';
X              yylval *= sign;
X              lptr--;
X              return (NUMBER);
X
X         } else if (isalpha(c)) {
X              p = idbuf;
X              while (isalpha(c = *lptr++) || c=='.')
X                   if (p < &idbuf[sizeof(idbuf)-1]) *p++ = c;
X              *p = '\0';
X              lptr--;
X              return (lookup(idbuf));
X         }
X
X         else if (c == '(') {
X              pcnt = 0;
X              do {
X                   c = *lptr++;
X                   if (c == '\0') return(c);
X                   else if (c == '(') pcnt++;
X                   else if (c == ')') pcnt--;
X              } while (pcnt > 0);
X         }
X
X         else return (*lptr++);
X    }
X}
X
Xstruct table {
X    char *name;
X    int type, value;
X};
X
Xstatic struct table mdtab[] = {
X    {"january", MONTH, 1},
X    {"february", MONTH, 2},
```

```
X    {"march", MONTH, 3},
X    {"april", MONTH, 4},
X    {"may", MONTH, 5},
X    {"june", MONTH, 6},
X    {"july", MONTH, 7},
X    {"august", MONTH, 8},
X    {"september", MONTH, 9},
X    {"sept", MONTH, 9},
X    {"october", MONTH, 10},
X    {"november", MONTH, 11},
X    {"december", MONTH, 12},
X
X    {"sunday", DAY, 0},
X    {"monday", DAY, 1},
X    {"tuesday", DAY, 2},
X    {"tues", DAY, 2},
X    {"wednesday", DAY, 3},
X    {"wednes", DAY, 3},
X    {"thursday", DAY, 4},
X    {"thur", DAY, 4},
X    {"thurs", DAY, 4},
X    {"friday", DAY, 5},
X    {"saturday", DAY, 6},
X    {0, 0, 0}};
X
X#define HRS *60
X#define HALFHR 30
X
Xstatic struct table mztab[] = {
X    {"a.m.", MERIDIAN, AM},
X    {"am", MERIDIAN, AM},
X    {"p.m.", MERIDIAN, PM},
X    {"pm", MERIDIAN, PM},
X    {"nst", ZONE, 3 HRS + HALFHR},        /* Newfoundland */
X    {"n.s.t.", ZONE, 3 HRS + HALFHR},
X    {"ast", ZONE, 4 HRS},        /* Atlantic */
X    {"a.s.t.", ZONE, 4 HRS},
X    {"adt", DAYZONE, 4 HRS},
X    {"a.d.t.", DAYZONE, 4 HRS},
X    {"est", ZONE, 5 HRS},        /* Eastern */
X    {"e.s.t.", ZONE, 5 HRS},
X    {"edt", DAYZONE, 5 HRS},
X    {"e.d.t.", DAYZONE, 5 HRS},
X    {"cst", ZONE, 6 HRS},        /* Central */
X    {"c.s.t.", ZONE, 6 HRS},
X    {"cdt", DAYZONE, 6 HRS},
X    {"c.d.t.", DAYZONE, 6 HRS},
X    {"mst", ZONE, 7 HRS},        /* Mountain */
X    {"m.s.t.", ZONE, 7 HRS},
X    {"mdt", DAYZONE, 7 HRS},
X    {"m.d.t.", DAYZONE, 7 HRS},
X    {"pst", ZONE, 8 HRS},        /* Pacific */
X    {"p.s.t.", ZONE, 8 HRS},
```

```
X    {"pdt", DAYZONE, 8 HRS},
X    {"p.d.t.", DAYZONE, 8 HRS},
X    {"yst", ZONE, 9 HRS},       /* Yukon */
X    {"y.s.t.", ZONE, 9 HRS},
X    {"ydt", DAYZONE, 9 HRS},
X    {"y.d.t.", DAYZONE, 9 HRS},
X    {"hst", ZONE, 10 HRS},      /* Hawaii */
X    {"h.s.t.", ZONE, 10 HRS},
X    {"hdt", DAYZONE, 10 HRS},
X    {"h.d.t.", DAYZONE, 10 HRS},
X
X    {"gmt", ZONE, 0 HRS},
X    {"g.m.t.", ZONE, 0 HRS},
X    {"bst", DAYZONE, 0 HRS},         /* British Summer Time */
X    {"b.s.t.", DAYZONE, 0 HRS},
X    {"eet", ZONE, 0 HRS},       /* European Eastern Time */
X    {"e.e.t.", ZONE, 0 HRS},
X    {"eest", DAYZONE, 0 HRS},   /* European Eastern Summer Time */
X    {"e.e.s.t.", DAYZONE, 0 HRS},
X    {"met", ZONE, -1 HRS},      /* Middle European Time */
X    {"m.e.t.", ZONE, -1 HRS},
X    {"mest", DAYZONE, -1 HRS},  /* Middle European Summer Time */
X    {"m.e.s.t.", DAYZONE, -1 HRS},
X    {"wet", ZONE, -2 HRS },     /* Western European Time */
X    {"w.e.t.", ZONE, -2 HRS },
X    {"west", DAYZONE, -2 HRS},  /* Western European Summer Time */
X    {"w.e.s.t.", DAYZONE, -2 HRS},
X
X    {"jst", ZONE, -9 HRS},      /* Japan Standard Time */
X    {"j.s.t.", ZONE, -9 HRS},   /* Japan Standard Time */
X                         /* No daylight savings time */
X
X    {"aest", ZONE, -10 HRS},    /* Australian Eastern Time */
X    {"a.e.s.t.", ZONE, -10 HRS},
X    {"aesst", DAYZONE, -10 HRS},     /* Australian Eastern Summer Time */
X    {"a.e.s.s.t.", DAYZONE, -10 HRS},
X    {"acst", ZONE, -(9 HRS + HALFHR)},   /* Australian Central Time */
X    {"a.c.s.t.", ZONE, -(9 HRS + HALFHR)},
X    {"acsst", DAYZONE, -(9 HRS + HALFHR)},    /* Australian Central Summer */
X    {"a.c.s.s.t.", DAYZONE, -(9 HRS + HALFHR)},
X    {"awst", ZONE, -8 HRS},     /* Australian Western Time */
X    {"a.w.s.t.", ZONE, -8 HRS}, /* (no daylight time there, I'm told */
X    {0, 0, 0}};
X
Xstatic struct table unittb[] = {
X    {"year", MUNIT, 12},
X    {"month", MUNIT, 1},
X    {"fortnight", UNIT, 14*24*60},
X    {"week", UNIT, 7*24*60},
X    {"day", UNIT, 1*24*60},
X    {"hour", UNIT, 60},
X    {"minute", UNIT, 1},
X    {"min", UNIT, 1},
```

```
X    {"second", SUNIT, 1},
X    {"sec", SUNIT, 1},
X    {0, 0, 0}};
X
Xstatic struct table othertb[] = {
X    {"tomorrow", UNIT, 1*24*60},
X    {"yesterday", UNIT, -1*24*60},
X    {"today", UNIT, 0},
X    {"now", UNIT, 0},
X    {"last", NUMBER, -1},
X    {"this", UNIT, 0},
X    {"next", NUMBER, 2},
X    {"first", NUMBER, 1},
X    /* {"second", NUMBER, 2}, */
X    {"third", NUMBER, 3},
X    {"fourth", NUMBER, 4},
X    {"fifth", NUMBER, 5},
X    {"sixth", NUMBER, 6},
X    {"seventh", NUMBER, 7},
X    {"eigth", NUMBER, 8},
X    {"ninth", NUMBER, 9},
X    {"tenth", NUMBER, 10},
X    {"eleventh", NUMBER, 11},
X    {"twelfth", NUMBER, 12},
X    {"ago", AGO, 1},
X    {0, 0, 0}};
X
Xstatic struct table milzone[] = {
X    {"a", ZONE, 1 HRS},
X    {"b", ZONE, 2 HRS},
X    {"c", ZONE, 3 HRS},
X    {"d", ZONE, 4 HRS},
X    {"e", ZONE, 5 HRS},
X    {"f", ZONE, 6 HRS},
X    {"g", ZONE, 7 HRS},
X    {"h", ZONE, 8 HRS},
X    {"i", ZONE, 9 HRS},
X    {"k", ZONE, 10 HRS},
X    {"l", ZONE, 11 HRS},
X    {"m", ZONE, 12 HRS},
X    {"n", ZONE, -1 HRS},
X    {"o", ZONE, -2 HRS},
X    {"p", ZONE, -3 HRS},
X    {"q", ZONE, -4 HRS},
X    {"r", ZONE, -5 HRS},
X    {"s", ZONE, -6 HRS},
X    {"t", ZONE, -7 HRS},
X    {"u", ZONE, -8 HRS},
X    {"v", ZONE, -9 HRS},
X    {"w", ZONE, -10 HRS},
X    {"x", ZONE, -11 HRS},
X    {"y", ZONE, -12 HRS},
X    {"z", ZONE, 0 HRS},
```

```
X    {0, 0, 0}};
X
Xstatic
Xlookup(id)
Xchar *id;
X{
X#define gotit (yylval=i->value,  i->type)
X
X    char idvar[128];
X    register char *j, *k;
X    register struct table *i;
X    int abbrev;
X
X    (void) strcpy(idvar, id);
X    j = idvar;
X    k = id - 1;
X    while (*++k)
X         *j++ = isupper(*k) ? tolower(*k) : *k;
X    *j = '\0';
X
X    if (strlen(idvar) == 3)
X         abbrev = 1;
X    else
X         if (strlen(idvar) == 4 && idvar[3] == '.') {
X              abbrev = 1;
X              idvar[3] = '\0';
X         }
X    else
X         abbrev = 0;
X
X    for (i = mdtab; i->name; i++) {
X         k = idvar;
X         for (j = i->name; *j++ == *k++;) {
X              if (abbrev && j == i->name+3)
X                   return gotit;
X              if (j[-1] == 0)
X                   return gotit;
X         }
X    }
X
X    for (i = mztab; i->name; i++)
X         if (strcmp(i->name, idvar) == 0)
X              return gotit;
X
X    for (i=mztab; i->name; i++)
X         if (strcmp(i->name, idvar) == 0)
X              return gotit;
X
X    for (i=unittb; i->name; i++)
X         if (strcmp(i->name, idvar) == 0)
X              return gotit;
X
X    if (idvar[strlen(idvar)-1] == 's')
```

```
X         idvar[strlen(idvar)-1] = '\0';
X
X    for (i=unittb; i->name; i++)
X         if (strcmp(i->name, idvar) == 0)
X              return gotit;
X
X    for (i = othertb; i->name; i++)
X         if (strcmp(i->name, idvar) == 0)
X              return gotit;
X
X    if (strlen(idvar) == 1 && isalpha(*idvar)) {
X         for (i = milzone; i->name; i++)
X              if (strcmp(i->name, idvar) == 0)
X                   return gotit;
X    }
X
X    return ID;
X}
X
Xtime_t
Xgetdate(p, now)
Xchar *p;
Xstruct timeb *now;
X{
X#define mcheck(f) if (f>1) err++
X    time_t monthadd();
X    int err;
X    struct tm *lt;
X    struct timeb ftz;
X
X    time_t sdate, tod;
X
X    lptr = p;
X    if (now == ((struct timeb *) NULL)) {
X         now = &ftz;
X         ftime(&ftz);
X    }
X    lt = localtime(&now->time);
X    year = lt->tm_year;
X    month = lt->tm_mon+1;
X    day = lt->tm_mday;
X    relsec = 0; relmonth = 0;
X    timeflag=zoneflag=dateflag=dayflag=relflag=0;
X    ourzone = now->timezone;
X    day_light = MAYBE;
X    hh = mm = ss = 0;
X    merid = 24;
X
X    if (err = yyparse()) return (-1);
X
X    mcheck(timeflag);
X    mcheck(zoneflag);
X    mcheck(dateflag);
```

```
X    mcheck(dayflag);
X
X    if (err) return (-1);
X    if (dateflag || timeflag || dayflag) {
X         sdate = dateconv(month,day,year,hh,mm,ss,merid,ourzone,day_light);
X         if (sdate < 0) return -1;
X    }
X    else {
X         sdate = now->time;
X         if (relflag == 0)
X              sdate -= (lt->tm_sec + lt->tm_min*60 +
X                   lt->tm_hour*(60L*60L));
X    }
X
X    sdate += relsec;
X    sdate += monthadd(sdate, relmonth);
X
X    if (dayflag && !dateflag) {
X         tod = dayconv(dayord, dayreq, sdate);
X         sdate += tod;
X    }
X
X    /*
X    ** Have to do *something* with a legitimate -1 so it's distinguishable
X    ** from the error return value.  (Alternately could set errno on error.)
X    */
X    return (sdate == -1) ? 0 : sdate;
X}
X
Xstatic
Xyyerror(s)
Xchar *s;
X{
X}
X
X#ifdef USG
Xftime(timeb)
Xstruct timeb *timeb;
X{
X    extern long timezone;
X    extern int daylight;
X
X    time(&(timeb->time));
X    timeb->millitm=0;
X    timeb->timezone=timezone;
X    timeb->dstflag=daylight;
X}
X#endif
SHAR_EOF
chmod 0644 GETDATE.Y || echo "restore of GETDATE.Y fails"
echo "x - extracting GETOLDOP.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > GETOLDOP.C &&
X/*
```

```
X
X    Copyright (C) 1988 Free Software Foundation
X
XGNU tar is distributed in the hope that it will be useful, but WITHOUT ANY
XWARRANTY.  No author or distributor accepts responsibility to anyone
Xfor the consequences of using it or for whether it serves any
Xparticular purpose or works at all, unless he says so in writing.
XRefer to the GNU tar General Public License for full details.
X
XEveryone is granted permission to copy, modify and redistribute GNU tar,
Xbut only under the conditions described in the GNU tar General Public
XLicense.  A copy of this license is supposed to have been given to you
Xalong with GNU tar so you can know your rights and responsibilities.  It
Xshould be in a file named COPYING.  Among other things, the copyright
Xnotice and this notice must be preserved on all copies.
X
XIn other words, go ahead and share GNU tar, but don't try to stop
Xanyone else from sharing it farther.  Help stamp out software hoarding!
X*/
X
X/*
X * Plug-compatible replacement for getopt() for parsing tar-like
X * arguments.  If the first argument begins with "-", it uses getopt;
X * otherwise, it uses the old rules used by tar, dump, and ps.
X *
X * Written 25 August 1985 by John Gilmore (ihnp4!hoptoad!gnu)
X *
X * @(#)getoldopt.c 1.4 2/4/86 - gnu
X */
X
X#include <stdio.h>
X
X
Xint
Xgetoldopt(argc, argv, optstring)
X    int  argc;
X    char **argv;
X    char *optstring;
X{
X    extern char   *optarg; /* Points to next arg */
X    extern int    optind;       /* Global argv index */
X    static char   *key;         /* Points to next keyletter */
X    static char   use_getopt;   /* !=0 if argv[1][0] was '-' */
X    extern char   *index();
X    char     c;
X    char     *place;
X
X    optarg = NULL;
X
X    if (key == NULL) {     /* First time */
X         if (argc < 2) return EOF;
X         key = argv[1];
X         if (*key == '-')
```

```
X             use_getopt++;
X        else
X             optind = 2;
X    }
X
X    if (use_getopt)
X         return getopt(argc, argv, optstring);
X
X    c = *key++;
X    if (c == '\0') {
X         key--;
X         return EOF;
X    }
X    place = index(optstring, c);
X
X    if (place == NULL || c == ':') {
X         msg("unknown option %c", c);
X         return('?');
X    }
X
X    place++;
X    if (*place == ':') {
X         if (optind < argc) {
X              optarg = argv[optind];
X              optind++;
X         } else {
X              msg("%c argument missing", c);
X              return('?');
X         }
X    }
X
X    return(c);
X}
SHAR_EOF
chmod 0644 GETOLDOP.C || echo "restore of GETOLDOP.C fails"
echo "x - extracting LIST.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > LIST.C &&
X/*
X
X    Copyright (C) 1988 Free Software Foundation
X
XGNU tar is distributed in the hope that it will be useful, but WITHOUT ANY
XWARRANTY.  No author or distributor accepts responsibility to anyone
Xfor the consequences of using it or for whether it serves any
Xparticular purpose or works at all, unless he says so in writing.
XRefer to the GNU tar General Public License for full details.
X
XEveryone is granted permission to copy, modify and redistribute GNU tar,
Xbut only under the conditions described in the GNU tar General Public
XLicense.  A copy of this license is supposed to have been given to you
Xalong with GNU tar so you can know your rights and responsibilities.  It
Xshould be in a file named COPYING.  Among other things, the copyright
Xnotice and this notice must be preserved on all copies.
```

```
X
XIn other words, go ahead and share GNU tar, but don't try to stop
Xanyone else from sharing it farther.  Help stamp out software hoarding!
X*/
X
X/*
X * List a tar archive.
X *
X * Also includes support routines for reading a tar archive.
X *
X * this version written 26 Aug 1985 by John Gilmore (ihnp4!hoptoad!gnu).
X *
X * @(#)list.c 1.31 11/5/87 - gnu
X */
X#include <stdio.h>
X#include <ctype.h>
X#ifdef atarist
X#include <types.h>
X#include <stat.h>
X#else
X#include <sys/types.h>
X#include <sys/stat.h>
X#endif
X
X#ifndef MSDOS
X#include <sys/file.h>
X#endif  /* MSDOS */
X
X#ifdef USG
X#include <sys/sysmacros.h> /* major() and minor() defined here */
X#endif
X
Xchar *ctime();                  /* From libc.a */
X
X#define isodigit(c)   ( ((c) >= '0') && ((c) <= '7') )
X
X#include "tar.h"
X#include "port.h"
X
Xlong from_oct();           /* Decode octal number */
Xvoid demode();                  /* Print file mode */
X
Xunion record *head;             /* Points to current archive header */
Xstruct stat hstat;              /* Stat struct corresponding */
Xint head_standard;              /* Tape header is in ANSI format */
X
Xvoid print_header();
Xvoid skip_file();
X
Xextern char *quote_copy_string();
X#ifdef atarist
Xextern FILE *msg_file;
X#endif
```

```
X
X/*
X * Main loop for reading an archive.
X */
Xvoid
Xread_and(do_something)
X    void (*do_something)();
X{
X    int status = 3;             /* Initial status at start of archive */
X    int prev_status;
X    extern time_t new_time;
X
X    name_gather();              /* Gather all the names */
X    open_archive(1);       /* Open for reading */
X
X    for(;;) {
X         prev_status = status;
X         status = read_header();
X         switch (status) {
X
X         case 1:            /* Valid header */
X              /* We should decode next field (mode) first... */
X              /* Ensure incoming names are null terminated. */
X              head->header.name[NAMSIZ-1] = '\0';
X
X              if (!name_match(head->header.name) || (f_new_files &&
hstat.st_mtime<new_time)) {
X                   /* Skip past it in the archive */
X                   userec(head);
X                   /* Skip to the next header on the archive */
X                   if(head->header.linkflag!='5')
X                        skip_file((long)hstat.st_size);
X                   continue;
X              }
X
X              (*do_something)();
X              continue;
X
X              /*
X               * If the previous header was good, tell them
X               * that we are skipping bad ones.
X               */
X         case 0:            /* Invalid header */
X              userec(head);
X              switch (prev_status) {
X              case 3:       /* Error on first record */
X                   msg("Hmm, this doesn't look like a tar archive.");
X                   /* FALL THRU */
X              case 2:       /* Error after record of zeroes */
X              case 1:       /* Error after header rec */
X                   msg("Skipping to next file header...\n");
X              case 0:       /* Error after error */
X                   break;
```

```
X             }
X             continue;
X
X        case 2:            /* Record of zeroes */
X             userec(head);
X             status = prev_status;  /* If error after 0's */
X             if (f_ignorez)
X                  continue;
X             /* FALL THRU */
X        case EOF:     /* End of archive */
X             break;
X        }
X        break;
X    };
X
X    close_archive();
X    names_notfound();       /* Print names not found */
X}
X
X
X/*
X * Print a header record, based on tar options.
X */
Xvoid
Xlist_archive()
X{
X    extern FILE *msg_file;
X    extern char *save_name;
X
X    /* Save the record */
X    saverec(&head);
X
X    /* Print the header record */
X    if (f_verbose) {
X         if (f_verbose > 1)
X              decode_header(head, &hstat, &head_standard, 0);
X         print_header();
X    }
X
X    if(f_gnudump && head->header.linkflag==LF_DUMPDIR) {
X         size_t   size, written, check;
X         char *data;
X         extern int errno;
X         extern long save_totsize;
X         extern long save_sizeleft;
X
X         userec(head);
X         if(f_multivol) {
X              save_name = head->header.name;
X              save_totsize=hstat.st_size;
X         }
X         for(size = hstat.st_size;size>0;size-=written) {
X              if(f_multivol)
```

```
X                 save_sizeleft=size;
X             data = findrec()->charptr;
X             if(data==NULL) {
X                  msg("EOF in archive file?");
X                  break;
X             }
X             written = endofrecs()->charptr - data;
X             if(written>size)
X                  written=size;
X             errno=0;
X             check=fwrite(data,sizeof(char), written, msg_file);
X             userec((union record *)(data+written - 1));
X             if(check!=written) {
X                  msg_perror("only wrote %ld of %ld bytes to file %s",check,
written,head->header.name);
X                  skip_file((long)(size)-written);
X                  break;
X             }
X        }
X        if(f_multivol)
X             save_name = 0;
X        saverec((union record **) 0);    /* Unsave it */
X        fputc('\n',msg_file);
X        return;
X
X    }
X    saverec((union record **) 0);    /* Unsave it */
X    /* Skip past it in the archive */
X    userec(head);
X
X    if(f_multivol)
X         save_name=head->header.name;
X    /* Skip to the next header on the archive */
X    skip_file((long)hstat.st_size);
X    if(f_multivol)
X         save_name = 0;
X}
X
X
X/*
X * Read a record that's supposed to be a header record.
X * Return its address in "head", and if it is good, the file's
X * size in hstat.st_size.
X *
X * Return 1 for success, 0 if the checksum is bad, EOF on eof,
X * 2 for a record full of zeros (EOF marker).
X *
X * You must always userec(head) to skip past the header which this
X * routine reads.
X */
Xint
Xread_header()
X{
```

```
X    register int  i;
X    register long sum, recsum;
X    register char *p;
X    register union record *header;
X    long from_oct();
X
X    header = findrec();
X    head = header;          /* This is our current header */
X    if (NULL == header) return EOF;
X
X    recsum = from_oct(8,  header->header.chksum);
X
X    sum = 0;
X    p = header->charptr;
X    for (i = sizeof(*header); --i >= 0;) {
X         /*
X          * We can't use unsigned char here because of old compilers,
X          * e.g. V7.
X          */
X         sum += 0xFF & *p++;
X    }
X
X    /* Adjust checksum to count the "chksum" field as blanks. */
X    for (i = sizeof(header->header.chksum); --i >= 0;)
X         sum -= 0xFF & header->header.chksum[i];
X    sum += ' '* sizeof header->header.chksum;
X
X    if (sum == recsum) {
X         /*
X          * Good record.  Decode file size and return.
X          */
X         if (header->header.linkflag == LF_LINK)
X              hstat.st_size = 0; /* Links 0 size on tape */
X         else
X              hstat.st_size = from_oct(1+12, header->header.size);
X         return 1;
X    }
X
X    if (sum == 8*' ') {
X         /*
X          * This is a zeroed record...whole record is 0's except
X          * for the 8 blanks we faked for the checksum field.
X          */
X         return 2;
X    }
X
X    return 0;
X}
X
X
X/*
X * Decode things from a file header record into a "struct stat".
X * Also set "*stdp" to !=0 or ==0 depending whether header record is "Unix
```

```
X * Standard" tar format or regular old tar format.
X *
X * read_header() has already decoded the checksum and length, so we don't.
X *
X * If wantug != 0, we want the uid/group info decoded from Unix Standard
X * tapes (for extraction).  If == 0, we are just printing anyway, so save time.
X *
X * decode_header should NOT be called twice for the same record, since the
X * two calls might use different "wantug" values and thus might end up with
X * different uid/gid for the two calls.  If anybody wants the uid/gid they
X * should decode it first, and other callers should decode it without uid/gid
X * before calling a routine, e.g. print_header, that assumes decoded data.
X */
Xdecode_header(header, st, stdp, wantug)
X    register union record  *header;
X    register struct stat   *st;
X    int  *stdp;
X    int  wantug;
X{
X
X    long from_oct();
X
X    st->st_mode = from_oct(8,  header->header.mode);
X    st->st_mtime = from_oct(1+12, header->header.mtime);
X    if(f_gnudump) {
X         st->st_atime = from_oct(1+12, header->header.atime);
X         st->st_ctime = from_oct(1+12, header->header.ctime);
X    }
X
X    if (0==strcmp(header->header.magic, TMAGIC)) {
X         /* Unix Standard tar archive */
X         *stdp = 1;
X         if (wantug) {
X#ifdef NONAMES
X              st->st_uid = from_oct(8,  header->header.uid);
X              st->st_gid = from_oct(8,  header->header.gid);
X#else
X              st->st_uid = finduid(header->header.uname);
X              st->st_gid = findgid(header->header.gname);
X#endif
X         }
X         switch  (header->header.linkflag)
X         case LF_BLK: case LF_CHR:
X             st->st_rdev = makedev(from_oct(8, header->header.devmajor),
X                         from_oct(8, header->header.devminor));
X    } else {
X         /* Old fashioned tar archive */
X         *stdp = 0;
X         st->st_uid = from_oct(8,  header->header.uid);
X         st->st_gid = from_oct(8,  header->header.gid);
X         st->st_rdev = 0;
X    }
X}
```

```
X
X
X/*
X * Quick and dirty octal conversion.
X *
X * Result is -1 if the field is invalid (all blank, or nonoctal).
X */
Xlong
Xfrom_oct(digs, where)
X    register int  digs;
X    register char *where;
X{
X    register long value;
X
X    while (isspace(*where)) {        /* Skip spaces */
X         where++;
X         if (--digs <= 0)
X              return -1;         /* All blank field */
X    }
X    value = 0;
X    while (digs > 0 && isodigit(*where)) {    /* Scan til nonoctal */
X         value = (value << 3) | (*where++ - '0');
X         --digs;
X    }
X
X    if (digs > 0 && *where && !isspace(*where))
X         return -1;              /* Ended on non-space/nul */
X
X    return value;
X}
X
X
X/*
X * Actually print it.
X *
X * Plain and fancy file header block logging.
X * Non-verbose just prints the name, e.g. for "tar t" or "tar x".
X * This should just contain file names, so it can be fed back into tar
X * with xargs or the "-T" option.  The verbose option can give a bunch
X * of info, one line per file.  I doubt anybody tries to parse its
X * format, or if they do, they shouldn't.  Unix tar is pretty random here
X * anyway.
X *
X * Note that print_header uses the globals <head>, <hstat>, and
X * <head_standard>, which must be set up in advance.  This is not very clean
X * and should be cleaned up.  FIXME.
X */
X#define  UGSWIDTH 11       /* min width of User, group, size */
X#define  DATEWIDTH 19      /* Last mod date */
Xstatic int   ugswidth = UGSWIDTH;    /* Max width encountered so far */
X
Xvoid
Xprint_header()
```

```
X{
X    char modes[11];
X    char *timestamp;
X    char uform[11], gform[11];  /* These hold formatted ints */
X    char *user, *group;
X    char size[24];         /* Holds a formatted long or maj, min */
X    long longie;       /* To make ctime() call portable */
X    int  pad;
X    extern FILE *msg_file;
X    char *name;
X    extern long baserec;
X
X    if(f_sayblock)
X         fprintf(msg_file,"rec %10d: ",baserec + (ar_record - ar_block));
X    /* annofile(msg_file, (char *)NULL); */
X
X    if (f_verbose <= 1) {
X         /* Just the fax, mam. */
X         char *name;
X
X         name=quote_copy_string(head->header.name);
X         if(name==0)
X              name=head->header.name;
X         fprintf(msg_file, "%s\n", name);
X         if(name!=head->header.name)
X              free(name);
X         return;
X    } else {
X         /* File type and modes */
X         modes[0] = '?';
X         switch (head->header.linkflag) {
X         case LF_VOLHDR:
X              modes[0]='V';
X              break;
X
X         case LF_MULTIVOL:
X              modes[0]='M';
X              break;
X
X         case LF_NORMAL:
X         case LF_OLDNORMAL:
X         case LF_LINK:
X                   modes[0] = '-';
X                   if ('/' == head->header.name[strlen(head->header.name)-1])
X                        modes[0] = 'd';
X                   break;
X         case LF_DUMPDIR:modes[0] = 'd'; break;
X         case LF_DIR:  modes[0] = 'd'; break;
X         case LF_SYMLINK:modes[0] = 'l'; break;
X         case LF_BLK:  modes[0] = 'b'; break;
X         case LF_CHR:  modes[0] = 'c'; break;
X         case LF_FIFO: modes[0] = 'p'; break;
X         case LF_CONTIG:    modes[0] = 'C'; break;
```

```
X         }
X
X         demode((unsigned)hstat.st_mode, modes+1);
X
X         /* Timestamp */
X         longie = hstat.st_mtime;
X         timestamp = ctime(&longie);
X         timestamp[16] = '\0';
X         timestamp[24] = '\0';
X
X         /* User and group names */
X         if (*head->header.uname && head_standard) {
X              user  = head->header.uname;
X         } else {
X              user = uform;
X              (void)sprintf(uform, "%d", (int)hstat.st_uid);
X         }
X         if (*head->header.gname && head_standard) {
X              group = head->header.gname;
X         } else {
X              group = gform;
X              (void)sprintf(gform, "%d", (int)hstat.st_gid);
X         }
X
X         /* Format the file size or major/minor device numbers */
X         switch (head->header.linkflag) {
X         case LF_CHR:
X         case LF_BLK:
X              (void)sprintf(size, "%d,%d",
X                       major(hstat.st_rdev),
X                       minor(hstat.st_rdev));
X              break;
X
X         default:
X              (void)sprintf(size, "%ld", (long)hstat.st_size);
X         }
X
X         /* Figure out padding and print the whole line. */
X         pad = strlen(user) + strlen(group) + strlen(size) + 1;
X         if (pad > ugswidth) ugswidth = pad;
X
X         name = quote_copy_string(head->header.name);
X         if(!name)
X              name=head->header.name;
X         fprintf(msg_file, "%s %s/%s %*s%s %s %s %.*s",
X              modes,
X              user,
X              group,
X              ugswidth - pad,
X              "",
X              size,
X              timestamp+4, timestamp+20,
X              sizeof(head->header.name),
```

```
X             name);
X
X         if(name!=head->header.name)
X             free(name);
X         switch (head->header.linkflag) {
X         case LF_SYMLINK:
X             name=quote_copy_string(head->header.linkname);
X             if(!name)
X                  name=head->header.linkname;
X             fprintf(msg_file, " -> %s\n", name);
X             if(name!=head->header.linkname)
X                  free(name);
X             break;
X
X         case LF_LINK:
X             name=quote_copy_string(head->header.linkname);
X             if(!name)
X                  name=head->header.linkname;
X             fprintf(msg_file, " link to %s\n", head->header.linkname);
X             if(name!=head->header.linkname)
X                  free(name);
X             break;
X
X         default:
X             fprintf(msg_file, " unknown file type '%c'\n",
X                  head->header.linkflag);
X             break;
X
X         case LF_OLDNORMAL:
X         case LF_NORMAL:
X         case LF_CHR:
X         case LF_BLK:
X         case LF_DIR:
X         case LF_FIFO:
X         case LF_CONTIG:
X         case LF_DUMPDIR:
X             putc('\n', msg_file);
X             break;
X
X         case LF_VOLHDR:
X             fprintf(msg_file, "--Volume Header--\n");
X             break;
X
X         case LF_MULTIVOL:
X             fprintf(msg_file, "--Continued at byte %ld--\n",from_oct(1+12,head-
>header.offset));
X             break;
X         }
X    }
X}
X
X/*
X * Print a similar line when we make a directory automatically.
```

```
X */
Xvoid
Xpr_mkdir(pathname, length, mode)
X    char *pathname;
X    int length;
X    int mode;
X{
X    char modes[11];
X    char *name;
X    extern long baserec;
X
X    if (f_verbose > 1) {
X         /* File type and modes */
X         modes[0] = 'd';
X         demode((unsigned)mode, modes+1);
X
X         if(f_sayblock)
X              fprintf(msg_file,"rec %10d: ",baserec + (ar_record - ar_block));
X         /* annofile(msg_file, (char *)NULL); */
X         name=quote_copy_string(pathname);
X         if(!name)
X              name=pathname;
X         fprintf(msg_file, "%s %*s %.*s\n",
X              modes,
X              ugswidth+DATEWIDTH,
X              "Creating directory:",
X              length,
X              pathname);
X         if(name!=pathname)
X              free(name);
X    }
X}
X
X
X/*
X * Skip over <size> bytes of data in records in the archive.
X */
Xvoid
Xskip_file(size)
X    register long size;
X{
X    union record *x;
X    extern long save_totsize;
X    extern long save_sizeleft;
X
X    if(f_multivol) {
X         save_totsize=size;
X         save_sizeleft=size;
X    }
X
X    while (size > 0) {
X         x = findrec();
X         if (x == NULL) {   /* Check it... */
```

```
X             msg("Unexpected EOF on archive file");
X             exit(EX_BADARCH);
X         }
X         userec(x);
X         size -= RECORDSIZE;
X         if(f_multivol)
X             save_sizeleft-=RECORDSIZE;
X    }
X}
X
X
X/*
X * Decode the mode string from a stat entry into a 9-char string and a null.
X */
Xvoid
Xdemode(mode, string)
X    register unsigned mode;
X    register char *string;
X{
X    register unsigned mask;
X    register char *rwx = "rwxrwxrwx";
X
X    for (mask = 0400; mask != 0; mask >>= 1) {
X         if (mode & mask)
X             *string++ = *rwx++;
X         else {
X             *string++ = '-';
X             rwx++;
X         }
X    }
X
X    if (mode & S_ISUID)
X         if (string[-7] == 'x')
X             string[-7] = 's';
X         else
X             string[-7] = 'S';
X    if (mode & S_ISGID)
X         if (string[-4] == 'x')
X             string[-4] = 's';
X         else
X             string[-4] = 'S';
X    if (mode & S_ISVTX)
X         if (string[-1] == 'x')
X             string[-1] = 't';
X         else
X             string[-1] = 'T';
X    *string = '\0';
X}
SHAR_EOF
chmod 0644 LIST.C || echo "restore of LIST.C fails"
echo "x - extracting MAKEFILE (Text)"
sed 's/^X//' << 'SHAR_EOF' > MAKEFILE &&
X# Makefile for public domain tar program.
```

```
X# @(#)Makefile 1.30   87/11/11
X
X## GNU version
X#DEFS = -DBSD42
X#LOCAL_SRC =
X#LOCAL_OBJ =
X#LDFLAGS =
X#LIBS =  -lutils
X#LINT = lint
X#LINTFLAGS = -abchx
X#DEF_AR_FILE = \"/dev/rmt8\"
X#DEFBLOCKING = 20
X#O = o
X
X## Berserkeley version
X#DEFS = -DBSD42
X#LOCAL_SRC = getdate.y  rtape_lib.c
X#LOCAL_OBJ = getdate.$O rtape_lib.$O
X#LDFLAGS =
X#LIBS =
X#LINT = lint
X#LINTFLAGS = -abchx
X#DEF_AR_FILE = \"/dev/rmt8\"
X#DEFBLOCKING = 20
X#O = o
X
X# USG version
X#DEFS = -DUSG
X#LOCAL_SRC =  getdate.y rtape_lib.c
X#LOCAL_OBJ =  getdate.$O rtape_lib.$O
X#LDFLAGS =
X#LIBS = -lndir -lPW
X#LINT = lint
X#LINTFLAGS = -p
X#DEF_AR_FILE = \"/dev/rmt8\"
X#DEFBLOCKING = 20
X#O = o
X
X# UniSoft's Uniplus SVR2 with NFS
X#DEFS = -DUSG -DUNIPLUS -DNFS -DSVR2
X#LOCAL_SRC =  getdate.y rtape_lib.c
X#LOCAL_OBJ =  getdate.$O rtape_lib.$O
X#LDFLAGS =
X#LIBS = -lndir
X#LINT = lint
X#LINTFLAGS = -bx
X#DEF_AR_FILE = \"/dev/rmt8\"
X#DEFBLOCKING = 20
X#O = o
X
X# MASSCOMP version
X#CC = ucb cc
X#DEFS = -DBSD42
```

```
X#LOCAL_SRC =  getdate.y rtape_lib.c
X#LOCAL_OBJ =  getdate.$O rtape_lib.$O
X#LDFLAGS =
X#LIBS =
X#LINT = lint
X#LINTFLAGS = -bx
X#DEF_AR_FILE = \"/dev/rmt0\"
X#DEFBLOCKING = 20
X#O = o
X
X# (yuk) MS-DOS (Microsoft C) version
X#MODEL = S
X#DEFS = -DNONAMES -A$(MODEL) -nologo
X#LOCAL_SRC =  getdate.y rtape_lib.c
X#LOCAL_OBJ =  getdate.$O rtape_lib.$O
X#LDFLAGS =
X#LIBS = $(MODEL)dir.lib
X#LINT = $(CC)
X#LINTFLAGS = -W3
X#DEF_AR_FILE = \"tar.out\"
X#DEFBLOCKING = 20
X#O = obj
X
X# V7 version
X# Pick open3 emulation or nonexistence.  See open3.h, port.c.
X##DEFS = -DV7 -DEMUL_OPEN3 -Dvoid=int
X##DEFS = -DV7 -DNO_OPEN3 -Dvoid=int
X#LOCAL_SRC =  getdate.y rtape_lib.c
X#LOCAL_OBJ =  getdate.$O rtape_lib.$O
X#LDFLAGS =
X#LIBS = -lndir
X#LINT = lint
X#LINTFLAGS = -abchx
X#DEF_AR_FILE = \"/dev/rmt8\"
X#DEFBLOCKING = 20
X#O = o
X
X# Minix version
X# No lint, so no lintflags.  Default file is stdin/out.  (Minix "tar"
X# doesn't even take an "f" flag, it assumes argv[2] is the archive name!)
X# Minix "make" doesn't expand macros right, so Minix users will have
X# to expand CFLAGS, SRCS, O, etc by hand, or fix your make.  Not my problem!
X# You'll also need to come up with getopt() and ctime(), the directory
X# library, and a fixed doprintf() that handles %*s.  Put this stuff in
X# the "SUBSRC/SUBOBJ" macro below if you didn't put it in your C library.
X# Note that Minix "cc" produces ".s" files, not .o's, so O = s should be
X# set (except on Atari ST, of course -- why do we have all this trouble
X# with "portable" operating systems?? -- ERS).
X#
X# Pick open3 emulation or nonexistence.  See open3.h, port.c.
X#DEFS = -DV7 -DMINIX -DEMUL_OPEN3
X##DEFS = -DV7 -DMINIX -DNO_OPEN3
X#LOCAL_SRC =  getdate.y rtape_lib.c
```

```
X#LOCAL_OBJ =  getdate.$O rtape_lib.$O
X#LDFLAGS =
X#LIBS =
X#DEF_AR_FILE = \"-\"
X#DEFBLOCKING = 8  /* No good reason for this, change at will */
X#O = o
X#YACC = /usr/local/bin/bison -y
X
X# Xenix version
X#DEFS = -DUSG -DXENIX
X#LOCAL_SRC =  getdate.y rtape_lib.c
X#LOCAL_OBJ =  getdate.$O rtape_lib.$O
X#LDFLAGS =
X#LIBS = -lx
X#LINT = lint
X#LINTFLAGS = -p
X#DEF_AR_FILE = \"/dev/rmt8\"
X#DEFBLOCKING = 20
X#O = o
X
X# Atari ST GEMDOS version (GCC v1.34, with new library)
X# Mostly, GEMDOS is like MSDOS, so that's what we pretend to be.
X# Also, for simplicity's sake the dumb getdate in port.c is used
X# instead of the fancy one (I hacked the dumb one up a bit, so it
X# isn't quite as bad as it used to be -- ERS).
X#
XCC = d:\gnu\bin\gcc.ttp
XDEFS = -DNONAMES -DMSDOS -DNO_REMOTE
X#LOCAL_SRC =  getdate.y rtape_lib.c
X#LOCAL_OBJ =  getdate.$O rtape_lib.$O
XLOCAL_SRC =
XLOCAL_OBJ =
XLDFLAGS = -s
XLIBS =
XLINT =  $(CC)
XLINTFLAGS = -Wall
XDEF_AR_FILE = 0       # This means take name from argv[2].
XDEFBLOCKING = 2
XO = o
X
XCFLAGS = $(COPTS) $(ALLDEFS)
XALLDEFS = $(DEFS) \
X    -DDEF_AR_FILE=$(DEF_AR_FILE) \
X    -DDEFBLOCKING=$(DEFBLOCKING)
X# next line for Debugging
X#COPTS = -g
X# next line for Production
XCOPTS = -O
X
X# Add things here like getopt, readdir, etc that aren't in your
X# standard libraries.  (E.g. MSDOS needs getopt, msd_dir.c, msd_dir.obj)
XSUBSRC=
XSUBOBJ=
```

```
X
X# Destination directory and installation program for make install
XDESTDIR = /usr/local
XINSTALL = cp
XRM = rm -f
X
XSRC1 =  tar.c create.c extract.c buffer.c getoldopt.c update.c
XSRC2 =  version.c list.c names.c diffarch.c port.c wildmat.c
XSRC3 =  $(LOCAL_SRC) $(SUBSRC)
XSRCS =  $(SRC1) $(SRC2) $(SRC3)
XOBJ1 =  tar.$O create.$O extract.$O buffer.$O getoldopt.$O list.$O update.$O
XOBJ2 =  version.$O names.$O diffarch.$O port.$O wildmat.$O
XOBJ3 =  $(LOCAL_OBJ) $(SUBOBJ)
XOBJS =  $(OBJ1) $(OBJ2) $(OBJ3)
X# AUX =  README PORTING Makefile TODO tar.h port.h open3.h \
X#   msd_dir.h msd_dir.c
XAUX =   README COPYING Makefile tar.texinfo tar.h port.h open3.h rmt.h \
X    msd_dir.h msd_dir.c rtape_server.c rtape_lib.c getdate.y
X
X# rmt is no good on the ST
X# all:  tar rmt
Xall:    tar
X
Xtar: $(OBJS)
X    $(CC) $(LDFLAGS) -o tar $(COPTS) $(OBJS) $(LIBS)
X
Xrmt:    rtape_server.c
X    $(CC) $(CFLAGS) -o rmt rtape_server.c
X
X# command is too long for Messy-Dos (128 char line length limit) so
X# this kludge is used...
X#   @echo $(OBJ1) + > command
X#   @echo $(OBJ2) >> command
X#   link @command, $@,,$(LIBS) /NOI;
X#   @$(RM) command
X
Xinstall: all
X    $(RM) $(DESTDIR)/bin/tar
X    $(INSTALL) tar   $(DESTDIR)/bin/tar
X    $(INSTALL) tar.texinfo $(DESTDIR)/man/tar.texinfo
X    $(INSTALL) rmt /etc/rmt
X
Xlint:   $(SRCS)
X    $(LINT) $(LINTFLAGS) $(ALLDEFS) $(SRCS)
X
Xclean:
X    $(RM) errs $(OBJS) tar rmt
X
Xtar.shar: $(SRCS) $(AUX)
X    shar >tar.shar1 $(AUX)
X    shar >tar.shar2 $(SRC1)
X    shar >tar.shar3 $(SRC2)
X
```

```
Xdist: tar.tar tar.tar.Z
X
Xtar.tar: $(SRCS) $(AUX)
X    /bin/tar cf tar.tar $(AUX) $(SRCS)
X
Xtar.tar.Z: tar.tar
X    compress < tar.tar > tar.tar.Z
X#   /bin/tar cf - $(AUX) $(SRCS) | compress -v >tar.tar.Z
X
Xtar.zoo: $(SRCS) $(AUX)
X    zoo a tar $(AUX) $(SRCS)
X
X$(OBJS): tar.h port.h
SHAR_EOF
chmod 0644 MAKEFILE || echo "restore of MAKEFILE fails"
echo "x - extracting MSD_DIR.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > MSD_DIR.C &&
X/*
X * @(#)msd_dir.c 1.4 87/11/06   Public Domain.
X *
X *  A public domain implementation of BSD directory routines for
X *  MS-DOS.  Written by Michael Rendell ({uunet,utai}michael@garfield),
X *  August 1897
X */
X
X#include <sys/types.h>
X#include <sys/stat.h>
X#include <sys/dir.h>
X#include <malloc.h>
X#include <string.h>
X#include <dos.h>
X
X#ifndef  NULL
X# define NULL 0
X#endif   /* NULL */
X
X#ifndef  MAXPATHLEN
X# define MAXPATHLEN    255
X#endif   /* MAXPATHLEN */
X
X/* attribute stuff */
X#define  A_RONLY       0x01
X#define  A_HIDDEN  0x02
X#define  A_SYSTEM  0x04
X#define  A_LABEL       0x08
X#define  A_DIR         0x10
X#define  A_ARCHIVE 0x20
X
X/* dos call values */
X#define  DOSI_FINDF    0x4e
X#define  DOSI_FINDN    0x4f
X#define  DOSI_SDTA 0x1a
X
```

```
X#define  Newisnull(a, t)        ((a = (t *) malloc(sizeof(t))) == (t *) NULL)
X/* #define   ATTRIBUTES         (A_DIR | A_HIDDEN | A_SYSTEM) */
X#define ATTRIBUTES    (A_RONLY | A_SYSTEM | A_DIR)
X
X/* what find first/next calls look use */
Xtypedef struct {
X    char     d_buf[21];
X    char     d_attribute;
X    unsigned short    d_time;
X    unsigned short    d_date;
X    long     d_size;
X    char     d_name[13];
X} Dta_buf;
X
Xstatic   char *getdirent();
Xstatic   void setdta();
Xstatic   void free_dircontents();
X
Xstatic   Dta_buf       dtabuf;
Xstatic   Dta_buf       *dtapnt = &dtabuf;
Xstatic   union REGS    reg, nreg;
X
X#if defined(M_I86LM)
Xstatic   struct SREGS  sreg;
X#endif
X
XDIR *
Xopendir(name)
X    char *name;
X{
X    struct   stat     statb;
X    DIR           *dirp;
X    char          c;
X    char          *s;
X    struct _dircontents    *dp;
X    char          nbuf[MAXPATHLEN + 1];
X
X    if (stat(name, &statb) < 0 || (statb.st_mode & S_IFMT) != S_IFDIR)
X         return (DIR *) NULL;
X    if (Newisnull(dirp, DIR))
X         return (DIR *) NULL;
X    if (*name && (c = name[strlen(name) - 1]) != '\\' && c != '/')
X         (void) strcat(strcpy(nbuf, name), "\\*.*");
X    else
X         (void) strcat(strcpy(nbuf, name), "*.*");
X    dirp->dd_loc = 0;
X    setdta();
X    dirp->dd_contents = dirp->dd_cp = (struct _dircontents *) NULL;
X    if ((s = getdirent(nbuf)) == (char *) NULL)
X         return dirp;
X    do {
X         if (Newisnull(dp, struct _dircontents) || (dp->_d_entry =
X              malloc((unsigned) (strlen(s) + 1))) == (char *) NULL)
```

```
X        {
X             if (dp)
X                  free((char *) dp);
X             free_dircontents(dirp->dd_contents);
X             return (DIR *) NULL;
X        }
X        if (dirp->dd_contents)
X             dirp->dd_cp = dirp->dd_cp->_d_next = dp;
X        else
X             dirp->dd_contents = dirp->dd_cp = dp;
X        (void) strcpy(dp->_d_entry, s);
X        dp->_d_next = (struct _dircontents *) NULL;
X    } while ((s = getdirent((char *) NULL)) != (char *) NULL);
X    dirp->dd_cp = dirp->dd_contents;
X
X    return dirp;
X}
X
Xvoid
Xclosedir(dirp)
X    DIR  *dirp;
X{
X    free_dircontents(dirp->dd_contents);
X    free((char *) dirp);
X}
X
Xstruct direct     *
Xreaddir(dirp)
X    DIR  *dirp;
X{
X    static   struct direct dp;
X
X    if (dirp->dd_cp == (struct _dircontents *) NULL)
X         return (struct direct *) NULL;
X    dp.d_namlen = dp.d_reclen =
X         strlen(strcpy(dp.d_name, dirp->dd_cp->_d_entry));
X    strlwr(dp.d_name);      /* JF */
X    dp.d_ino = 0;
X    dirp->dd_cp = dirp->dd_cp->_d_next;
X    dirp->dd_loc++;
X
X    return &dp;
X}
X
Xvoid
Xseekdir(dirp, off)
X    DIR  *dirp;
X    long off;
X{
X    long          i = off;
X    struct _dircontents    *dp;
X
X    if (off < 0)
```

```
X        return;
X    for (dp = dirp->dd_contents ; --i >= 0 && dp ; dp = dp->_d_next)
X        ;
X    dirp->dd_loc = off - (i + 1);
X    dirp->dd_cp = dp;
X}
X
Xlong
Xtelldir(dirp)
X    DIR  *dirp;
X{
X    return dirp->dd_loc;
X}
X
Xstatic  void
Xfree_dircontents(dp)
X    struct   _dircontents  *dp;
X{
X    struct _dircontents    *odp;
X
X    while (dp) {
X        if (dp->_d_entry)
X            free(dp->_d_entry);
X        dp = (odp = dp)->_d_next;
X        free((char *) odp);
X    }
X}
X
Xstatic  char *
Xgetdirent(dir)
X    char *dir;
X{
X    if (dir != (char *) NULL) {      /* get first entry */
X        reg.h.ah = DOSI_FINDF;
X        reg.h.cl = ATTRIBUTES;
X#if defined(M_I86LM)
X        reg.x.dx = FP_OFF(dir);
X        sreg.ds = FP_SEG(dir);
X#else
X        reg.x.dx = (unsigned) dir;
X#endif
X    } else {                /* get next entry */
X        reg.h.ah = DOSI_FINDN;
X#if defined(M_I86LM)
X        reg.x.dx = FP_OFF(dtapnt);
X        sreg.ds = FP_SEG(dtapnt);
X#else
X        reg.x.dx = (unsigned) dtapnt;
X#endif
X    }
X#if defined(M_I86LM)
X    intdosx(&reg, &nreg, &sreg);
X#else
```

```
X    intdos(&reg, &nreg);
X#endif
X    if (nreg.x.cflag)
X         return (char *) NULL;
X
X    return dtabuf.d_name;
X}
X
Xstatic  void
Xsetdta()
X{
X    reg.h.ah = DOSI_SDTA;
X#if defined(M_I86LM)
X    reg.x.dx = FP_OFF(dtapnt);
X    sreg.ds = FP_SEG(dtapnt);
X    intdosx(&reg, &nreg, &sreg);
X#else
X    reg.x.dx = (int) dtapnt;
X    intdos(&reg, &nreg);
X#endif
X}
SHAR_EOF
chmod 0644 MSD_DIR.C || echo "restore of MSD_DIR.C fails"
echo "x - extracting MSD_DIR.H (Text)"
sed 's/^X//' << 'SHAR_EOF' > MSD_DIR.H &&
X/*
X * @(#)msd_dir.h 1.4 87/11/06   Public Domain.
X *
X *  A public domain implementation of BSD directory routines for
X *  MS-DOS.  Written by Michael Rendell ({uunet,utai}michael@garfield),
X *  August 1897
X */
X
X#define  rewinddir(dirp)    seekdir(dirp, 0L)
X
X#define  MAXNAMLEN 12
X
Xstruct direct {
X    ino_t    d_ino;             /* a bit of a farce */
X    int  d_reclen;     /* more farce */
X    int  d_namlen;     /* length of d_name */
X    char d_name[MAXNAMLEN + 1];      /* garentee null termination */
X};
X
Xstruct _dircontents {
X    char *_d_entry;
X    struct _dircontents    *_d_next;
X};
X
Xtypedef struct _dirdesc {
X    int      dd_id;    /* uniquely identify each open directory */
X    long     dd_loc;   /* where we are in directory entry is this */
X    struct _dircontents    *dd_contents; /* pointer to contents of dir */
```

```
X    struct _dircontents    *dd_cp;   /* pointer to current position */
X} DIR;
X
Xextern  DIR      *opendir();
Xextern  struct direct *readdir();
Xextern  void     seekdir();
Xextern  long     telldir();
Xextern  void     closedir();
SHAR_EOF
chmod 0644 MSD_DIR.H || echo "restore of MSD_DIR.H fails"
echo "x - extracting NAMES.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > NAMES.C &&
X/*
X
X    Copyright (C) 1988 Free Software Foundation
X
XGNU tar is distributed in the hope that it will be useful, but WITHOUT ANY
XWARRANTY.  No author or distributor accepts responsibility to anyone
Xfor the consequences of using it or for whether it serves any
Xparticular purpose or works at all, unless he says so in writing.
SHAR_EOF
echo "End of part 3"
echo "File NAMES.C is continued in part 4"
echo "4" > s2_seq_.tmp
exit 0


From 2920544039838250363
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,cfc3396ca74a3445,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-19 14:43:14 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
agate!apple!sun-barr!newstop!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i050: gnutar -- GNU's version of UNIX "tar"  part04/08
Keywords: shar, C
Message-ID: <33890@grapevine.uucp>
Date: 19 Jul 89 21:43:14 GMT
Sender: news@grapevine.uucp
Lines: 1603
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: 7103_300@uwovax.uwo.ca (Eric R. Smith)
Posting-number: Volume 2, Issue 50
Archive-name: gnutar/part04

#!/bin/sh
# this is part 4 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file NAMES.C continued
#
```

```
CurArch=4
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file NAMES.C"
sed 's/^X//' << 'SHAR_EOF' >> NAMES.C
XRefer to the GNU tar General Public License for full details.
X
XEveryone is granted permission to copy, modify and redistribute GNU tar,
Xbut only under the conditions described in the GNU tar General Public
XLicense.  A copy of this license is supposed to have been given to you
Xalong with GNU tar so you can know your rights and responsibilities.  It
Xshould be in a file named COPYING.  Among other things, the copyright
Xnotice and this notice must be preserved on all copies.
X
XIn other words, go ahead and share GNU tar, but don't try to stop
Xanyone else from sharing it farther.  Help stamp out software hoarding!
X*/
X
X/*
X * Look up user and/or group names.
X *
X * This file should be modified for non-unix systems to do something
X * reasonable.
X *
X * @(#)names.c 1.3 10/30/87 - gnu
X */
X#ifdef atarist
X#include <types.h>
X#else
X#include <sys/types.h>
X#endif
X
X#include "tar.h"
X
Xextern  char *strncpy();
X
X#ifndef NONAMES
X/* Whole module goes away if NONAMES defined.  Otherwise... */
X#include <pwd.h>
X#include <grp.h>
X
Xstatic int   saveuid = -993;
Xstatic char  saveuname[TUNMLEN];
Xstatic int   my_uid = -993;
X
Xstatic int   savegid = -993;
```

```
Xstatic char  savegname[TGNMLEN];
Xstatic int   my_gid = -993;
X
X#define myuid     ( my_uid < 0? (my_uid = getuid()): my_uid )
X#define mygid     ( my_gid < 0? (my_gid = getgid()): my_gid )
X
X/*
X * Look up a user or group name from a uid/gid, maintaining a cache.
X * FIXME, for now it's a one-entry cache.
X * FIXME2, the "-993" is to reduce the chance of a hit on the first lookup.
X *
X * This is ifdef'd because on Suns, it drags in about 38K of "yellow
X * pages" code, roughly doubling the program size.  Thanks guys.
X */
Xvoid
Xfinduname(uname, uid)
X    char uname[TUNMLEN];
X    int  uid;
X{
X    struct passwd *pw;
X    extern struct passwd *getpwuid ();
X
X    if (uid != saveuid) {
X         saveuid = uid;
X         saveuname[0] = '\0';
X         pw = getpwuid(uid);
X         if (pw)
X              strncpy(saveuname, pw->pw_name, TUNMLEN);
X    }
X    strncpy(uname, saveuname, TUNMLEN);
X}
X
Xint
Xfinduid(uname)
X    char uname[TUNMLEN];
X{
X    struct passwd *pw;
X    extern struct passwd *getpwnam();
X
X    if (uname[0] != saveuname[0]     /* Quick test w/o proc call */
X        || 0!=strncmp(uname, saveuname, TUNMLEN)) {
X         strncpy(saveuname, uname, TUNMLEN);
X         pw = getpwnam(uname);
X         if (pw) {
X              saveuid = pw->pw_uid;
X         } else {
X              saveuid = myuid;
X         }
X    }
X    return saveuid;
X}
X
X
```

```
Xvoid
Xfindgname(gname, gid)
X    char gname[TGNMLEN];
X    int  gid;
X{
X    struct group  *gr;
X    extern struct group *getgrgid ();
X
X    if (gid != savegid) {
X         savegid = gid;
X         savegname[0] = '\0';
X         (void)setgrent();
X         gr = getgrgid(gid);
X         if (gr)
X              strncpy(savegname, gr->gr_name, TGNMLEN);
X    }
X    (void) strncpy(gname, savegname, TGNMLEN);
X}
X
X
Xint
Xfindgid(gname)
X    char gname[TUNMLEN];
X{
X    struct group  *gr;
X    extern struct group *getgrnam();
X
X    if (gname[0] != savegname[0]     /* Quick test w/o proc call */
X        || 0!=strncmp(gname, savegname, TUNMLEN)) {
X         strncpy(savegname, gname, TUNMLEN);
X         gr = getgrnam(gname);
X         if (gr) {
X              savegid = gr->gr_gid;
X         } else {
X              savegid = mygid;
X         }
X    }
X    return savegid;
X}
X#endif
SHAR_EOF
echo "File NAMES.C is complete"
chmod 0644 NAMES.C || echo "restore of NAMES.C fails"
echo "x - extracting OPEN3.H (Text)"
sed 's/^X//' << 'SHAR_EOF' > OPEN3.H &&
X/*
X
X    Copyright (C) 1988 Free Software Foundation
X
XGNU tar is distributed in the hope that it will be useful, but WITHOUT ANY
XWARRANTY.  No author or distributor accepts responsibility to anyone
Xfor the consequences of using it or for whether it serves any
Xparticular purpose or works at all, unless he says so in writing.
```

```
XRefer to the GNU tar General Public License for full details.
X
XEveryone is granted permission to copy, modify and redistribute GNU tar,
Xbut only under the conditions described in the GNU tar General Public
XLicense.  A copy of this license is supposed to have been given to you
Xalong with GNU tar so you can know your rights and responsibilities.  It
Xshould be in a file named COPYING.  Among other things, the copyright
Xnotice and this notice must be preserved on all copies.
X
XIn other words, go ahead and share GNU tar, but don't try to stop
Xanyone else from sharing it farther.  Help stamp out software hoarding!
X*/
X
X/*
X * @(#)open3.h 1.4 87/11/11
X *
X * open3.h -- #defines for the various flags for the Sys V style 3-argument
X * open() call.  On BSD or System 5, the system already has this in an
X * include file.  This file is needed for V7 and MINIX systems for the
X * benefit of open3() in port.c, a routine that emulates the 3-argument
X * call using system calls available on V7/MINIX.
X *
X * This file is needed by PD tar even if we aren't using the
X * emulator, since the #defines for O_WRONLY, etc. are used in
X * a couple of places besides the open() calls, (e.g. in the assignment
X * to openflag in extract.c).  We just #include this rather than
X * #ifdef them out.
X *
X * Written 6/10/87 by rmtodd@uokmax (Richard Todd).
X *
X * The names have been changed by John Gilmore, 31 July 1987, since
X * Richard called it "bsdopen", and really this change was introduced in
X * AT&T Unix systems before BSD picked it up.
X */
X
X/* Only one of the next three should be specified */
X#define O_RDONLY   0 /* only allow read */
X#define  O_WRONLY  1 /* only allow write */
X#define  O_RDWR         2 /* both are allowed */
X
X/* The rest of these can be OR-ed in to the above. */
X/*
X * O_NDELAY isn't implemented by the emulator.  It's only useful (to tar) on
X * systems that have named pipes anyway; it prevents tar's hanging by
X * opening a named pipe.  We #ifndef it because some systems already have
X * it defined.
X */
X#ifndef O_NDELAY
X#define O_NDELAY   4 /* don't block on opening devices that would
X                 * block on open -- ignored by emulator. */
X#endif
X#define O_CREAT        8 /* create file if needed */
X#define O_EXCL        16 /* file cannot already exist */
```

```
X#define O_TRUNC       32 /* truncate file on open */
X#define O_APPEND  64 /* always write at end of file -- ignored by emul */
X
X#ifdef EMUL_OPEN3
X/*
X * make emulation transparent to rest of file -- redirect all open() calls
X * to our routine
X */
X#define open open3
X#endif
SHAR_EOF
chmod 0644 OPEN3.H || echo "restore of OPEN3.H fails"
echo "x - extracting PORT.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > PORT.C &&
X/*
X
X    Copyright (C) 1988 Free Software Foundation
X
XGNU tar is distributed in the hope that it will be useful, but WITHOUT ANY
XWARRANTY.  No author or distributor accepts responsibility to anyone
Xfor the consequences of using it or for whether it serves any
Xparticular purpose or works at all, unless he says so in writing.
XRefer to the GNU tar General Public License for full details.
X
XEveryone is granted permission to copy, modify and redistribute GNU tar,
Xbut only under the conditions described in the GNU tar General Public
XLicense.  A copy of this license is supposed to have been given to you
Xalong with GNU tar so you can know your rights and responsibilities.  It
Xshould be in a file named COPYING.  Among other things, the copyright
Xnotice and this notice must be preserved on all copies.
X
XIn other words, go ahead and share GNU tar, but don't try to stop
Xanyone else from sharing it farther.  Help stamp out software hoarding!
X*/
X
X/*
X * @(#)port.c 1.15    87/11/05 by John Gilmore, 1986
X *
X * These are routines not available in all environments.
X *
X * I know this introduces an extra level of subroutine calls and is
X * slightly slower.  Frankly, my dear, I don't give a damn.  Let the
X * Missed-Em Vee losers suffer a little.  This software is proud to
X * have been written on a BSD system.
X */
X#include <stdio.h>
X#ifdef atarist
X#include <types.h>
X#include <time.h>
X#include <stat.h>
X#else
X#include <sys/types.h>
X#include <sys/stat.h>
```

```
X#endif
X#include <signal.h>
X#include <errno.h>
X
X#ifdef MSDOS
X#include <fcntl.h>
X#else
X#include <sys/file.h>
X#endif
X
X#include "tar.h"
X#include "port.h"
X
Xextern size_t strlen();
X
Xextern long baserec;
X/*
X * Some people (e.g. V7) don't have a #define for these.
X */
X#ifndef O_BINARY
X#define O_BINARY 0
X#endif
X#ifndef O_RDONLY
X#define O_RDONLY 0
X#endif
X#ifndef NULL
X#define NULL 0
X#endif
X
X/* JF: modified so all configuration information can appear here, instead of
X   being scattered through the file.  Add all the machine-dependent #ifdefs
X   here */
X#undef WANT_DUMB_GETDATE/* WANT_DUMB_GETDATE --> getdate() */
X#undef WANT_LSTAT /* WANT_LSTAT --> lstat() */
X#undef WANT_VALLOC     /* WANT_VALLOC --> valloc() */
X#undef WANT_MKDIR /* WANT_MKDIR --> mkdir() */
X#undef WANT_STRING     /* WANT_STRING --> index() bcopy() bzero() bcmp() */
X#undef WANT_BZERO /* WANT_BZERO --> bzero() bcmp() execlp() */
X              /* EMUL_OPEN3 --> open3() */
X#undef WANT_MKNOD /* WANT_MKNOD --> mknod() link() chown() geteuid() */
X#undef WANT_UTILS /* WANT_UTILS --> panic() ck_*() *_buffer()
X                merge_sort() quote_copy_string() un_quote_string() */
X#undef WANT_CK_PIPE    /* WANT_CK_PIPE --> ck_pipe() getwd() */
X#undef WANT_GETOPT     /* WANT_GETOPT --> getopt() optind optarg */
X#undef WANT_STRSTR     /* WANT_STRSTR --> strstr() */
X
X/* Define only ONE of these four . . . */
X#undef DOPRNT_MSG /* Define this one if you have _doprnt() and
X                no varargs support */
X#undef VARARGS_MSG     /* Define this one if you have varargs.h and
X                vfprintf() */
X#undef STDC_MSG        /* Define this one if you are using ANSI C and
X                and have vfprintf() */
```

```
X#undef LOSING_MSG /* Define this one if you don't have any of the
X                above */
X#ifdef USG
X#define WANT_STRING
X#define WANT_VALLOC
X#endif
X
X#ifdef MINIX
X#define WANT_BZERO
X#endif
X
X/* MSDOS stuff changed around for atarist by ERS */
X#ifdef MSDOS
X# define WANT_LSTAT
X# define WANT_MKNOD
X# define WANT_UTILS
X# define WANT_GETOPT
X# define WANT_VALLOC
X# ifdef atarist
X#  define WANT_DUMB_GETDATE
X#  define VARARGS_MSG
X# else
X#  define WANT_STRING
X# endif
X#else
X# define VARARGS_MSG
X# define WANT_UTILS
X# define WANT_CK_PIPE
X# define WANT_STRSTR
X# define WANT_GETOPT
X#endif
X
X/* End of system-dependent #ifdefs */
X
X#ifdef WANT_DUMB_GETDATE
X/* JF a getdate() routine takes a date/time/etc and turns it into a time_t */
X/* This one is a quick hack I wrote in about five minutes to see if the N
X   option works.  Someone should replace it with one that works */
X
X/* This getdate takes an arg of the form mm/dd/yyyy hh:mm:ss and turns it
X   into a time_t .  Its not well tested or anything. . .  */
X/* In general, you should use the getdate() supplied in getdate.y */
X
X/*#define OFF_FROM_GMT 18000        /* Change for your time zone! */
X#define OFF_FROM_GMT 0
X
Xtime_t
Xgetdate(str)
Xchar *str;
X{
X    int month,day,year,hour,minute,second;
X#ifdef atarist
X    struct tm t;
```

```
X    int  n;
X#else
X    time_t ret;
X    int  n;
X
X#define SECS_PER_YEAR (365L*SECS_PER_DAY)
X#define SECS_PER_LEAP_YEAR (366L*SECS_PER_DAY)
X
X#define SECS_PER_DAY (24L*60*60)
X    static int days_per_month[2][12] = {
X         31,28,31,30,31,30,31,31,30,31,30,31,
X         31,29,31,30,31,30,31,31,30,31,30,31
X    };
X
X    static int days_per_year[2]={365,366};
X#endif
X    month=day=year=hour=minute=second=0;
X    n=sscanf(str,"%d/%d/%d %d:%d:%d",&month,&day,&year,&hour,&minute,&second);
X    if(n<3)
X         return 0;
X    if(year<100)
X         year+=1900;
X    if(year<1970)
X         return 0;
X#ifdef atarist
X    t.tm_sec = second;
X    t.tm_min = minute;
X    t.tm_hour = hour;
X    t.tm_mday = day;
X    t.tm_mon = month - 1;
X    t.tm_year = year - 1900;
X    return mktime(&t);
X#else
X    ret=0;
X
X    ret+=OFF_FROM_GMT;
X
X    for(n=1970;n<year;n++)
X         if(n%4==0 && n%400!=0)
X              ret+=SECS_PER_LEAP_YEAR;
X         else
X              ret+=SECS_PER_YEAR;
X
X    month--;
X    for(n=0;n<month;n++) {
X         if(year%4==0 && year%400!=0)
X              ret+=SECS_PER_DAY*days_per_month[1][n];
X         else
X              ret+=SECS_PER_DAY*days_per_month[0][n];
X    }
X    ret+=SECS_PER_DAY*(day-1);
X    ret+=second+minute*60+hour*60*60;
X    return ret;
```

```
X#endif /* atarist */
X}
X#endif
X
X/*
X * lstat() is a stat() which does not follow symbolic links.
X * If there are no symbolic links, just use stat().
X */
X#ifdef WANT_LSTAT
Xint
Xlstat (path, buf)
X    char *path;
X    struct stat *buf;
X{
X    extern int stat ();
X    return (stat (path, buf));
X}
X#endif
X
X#ifdef WANT_VALLOC
X/*
X * valloc() does a malloc() on a page boundary.  On some systems,
X * this can make large block I/O more efficient.
X */
Xchar *
Xvalloc (size)
X    unsigned size;
X{
X    extern char *malloc ();
X    return (malloc (size));
X}
X#endif
X/*
X *                NMKDIR.C
X *
X * Written by Robert Rother, Mariah Corporation, August 1985.
X *
X * I wrote this out of shear disgust with myself because I couldn't
X * figure out how to do this in /bin/sh.
X *
X * If you want it, it's yours.  All I ask in return is that if you
X * figure out how to do this in a Bourne Shell script you send me
X * a copy.
X *                     sdcsvax!rmr or rmr@uscd
X*
X* Severely hacked over by John Gilmore to make a 4.2BSD compatible
X* subroutine.     11Mar86; hoptoad!gnu
X*
X* Modified by rmtodd@uokmax 6-28-87 -- when making an already existing dir,
X* subroutine didn't return EEXIST.  It does now.
X*/
X
X/*
```

```
X * Make a directory.  Compatible with the mkdir() system call on 4.2BSD.
X */
X#ifdef WANT_MKDIR
Xint
Xmkdir(dpath, dmode)
X    char *dpath;
X    int dmode;
X{
X    int cpid, status;
X    struct stat statbuf;
X    extern int errno;
X
X    if (stat(dpath,&statbuf) == 0) {
X         errno = EEXIST;        /* Stat worked, so it already exists */
X         return -1;
X    }
X
X    /* If stat fails for a reason other than non-existence, return error */
X    if (errno != ENOENT) return -1;
X
X    switch (cpid = fork()) {
X
X    case -1:           /* Error in fork() */
X         return(-1);        /* Errno is set already */
X
X    case 0:                 /* Child process */
X         /*
X          * Cheap hack to set mode of new directory.  Since this
X          * child process is going away anyway, we zap its umask.
X          * FIXME, this won't suffice to set SUID, SGID, etc. on this
X          * directory.  Does anybody care?
X          */
X         status = umask(0); /* Get current umask */
X         status = umask(status | (0777 & ~dmode)); /* Set for mkdir */
X         execl("/bin/mkdir", "mkdir", dpath, (char *)0);
X         _exit(-1);         /* Can't exec /bin/mkdir */
X
X    default:           /* Parent process */
X         while (cpid != wait(&status)) ;  /* Wait for kid to finish */
X    }
X
X    if (TERM_SIGNAL(status) != 0 || TERM_VALUE(status) != 0) {
X         errno = EIO;       /* We don't know why, but */
X         return -1;         /* /bin/mkdir failed */
X    }
X
X    return 0;
X}
X#endif
X
X#ifdef WANT_STRING
X/*
X * Translate V7 style into Sys V style.
```

```
X */
X#include <string.h>
X#include <memory.h>
X
Xchar *
Xindex (s, c)
X    char *s;
X    int c;
X{
X    return (strchr (s, c));
X}
X
Xchar *
Xrindex(s,c)
Xchar *s;
Xint c;
X{
X    return strrchr(s,c);
X}
X
Xvoid
Xbcopy (s1, s2, n)
X    char *s1, *s2;
X    int n;
X{
X    (void) memcpy (s2, s1, n);
X}
X
Xvoid
Xbzero (s1, n)
X    char *s1;
X    int n;
X{
X    (void) memset(s1, 0, n);
X}
X
Xint
Xbcmp(s1, s2, n)
X    char *s1, *s2;
X    int  n;
X{
X    return memcmp(s1, s2, n);
X}
X#endif
X
X#ifdef WANT_BZERO
X/* Minix has bcopy but not bzero, and no memset.  Thanks, Andy. */
Xvoid
Xbzero (s1, n)
X    register char *s1;
X    register int n;
X{
X    while (n--) *s1++ = '\0';
```

```
X}
X
X/* It also has no bcmp() */
Xint
Xbcmp (s1, s2, n)
X    register char *s1,*s2;
X    register int n;
X{
X    for ( ; n-- ; ++s1, ++s2) {
X         if (*s1 != *s2) return *s1 - *s2;
X    }
X    return 0;
X}
X
X/*
X * Groan, Minix doesn't have execlp either!
X *
X * execlp(file,arg0,arg1...argn,(char *)NULL)
X * exec a program, automatically searching for the program through
X * all the directories on the PATH.
X *
X * This version is naive about variable argument lists, it assumes
X * a straightforward C calling sequence.  If your system has odd stacks
X * *and* doesn't have execlp, YOU get to fix it.
X */
Xint
Xexeclp(filename, arg0)
X    char *filename, *arg0;
X{
X    register char *p, *path;
X    register char *fnbuffer;
X    char **argstart = &arg0;
X    struct stat statbuf;
X    extern char **environ;
X    extern int errno;
X    extern char *malloc(), *getenv(), *index();
X
X    if ((p = getenv("PATH")) == NULL) {
X         /* couldn't find path variable -- try to exec given filename */
X         return execve(filename, argstart, environ);
X    }
X
X    /*
X     * make a place to build the filename.  We malloc larger than we
X     * need, but we know it will fit in this.
X     */
X    fnbuffer = malloc( strlen(p) + 1 + strlen(filename) );
X    if (fnbuffer == NULL) {
X         errno = ENOMEM;
X         return -1;
X    }
X
X    /*
```

```
X     * try each component of the path to see if the file's there
X     * and executable.
X     */
X    for (path = p ; path ; path = p) {
X         /* construct full path name to try */
X         if ((p = index(path,':')) == NULL) {
X              strcpy(fnbuffer, path);
X         } else {
X              strncpy(fnbuffer, path, p-path);
X              fnbuffer[p-path] = '\0';
X              p++;      /* Skip : for next time */
X         }
X         if (strlen(fnbuffer) != 0)
X              strcat(fnbuffer,"/");
X         strcat(fnbuffer,filename);
X
X         /* check to see if file is there and is a normal file */
X         if (stat(fnbuffer, &statbuf) < 0) {
X              if (errno == ENOENT)
X                   continue; /* file not there,keep on looking */
X              else
X                   goto fail; /* failed for some reason, return */
X         }
X         if ( (statbuf.st_mode & S_IFMT) != S_IFREG) continue;
X
X         if (execve(fnbuffer, argstart, environ) < 0
X             && errno != ENOENT
X             && errno != ENOEXEC) {
X              /* failed, for some other reason besides "file
X               * not found" or "not a.out format"
X               */
X              goto fail;
X         }
X
X         /*
X          * If we got error ENOEXEC, the file is executable but is
X          * not an object file.  Try to execute it as a shell script,
X          * returning error if we can't execute /bin/sh.
X          *
X          * FIXME, this code is broken in several ways.  Shell
X          * scripts should not in general be executed by the user's
X          * SHELL variable program.  On more mature systems, the
X          * script can specify with #!/bin/whatever.  Also, this
X          * code clobbers argstart[-1] if the exec of the shell
X          * fails.
X          */
X         if (errno == ENOEXEC) {
X              char *shell;
X
X              /* Try to execute command "sh arg0 arg1 ..." */
X              if ((shell = getenv("SHELL")) == NULL)
X                   shell = "/bin/sh";
X              argstart[-1] = shell;
```

```
X             argstart[0] = fnbuffer;
X             execve(shell, &argstart[-1], environ);
X             goto fail;    /* Exec didn't work */
X         }
X
X         /*
X          * If we succeeded, the execve() doesn't return, so we
X          * can only be here is if the file hasn't been found yet.
X          * Try the next place on the path.
X          */
X    }
X
X    /* all attempts failed to locate the file.  Give up. */
X    errno = ENOENT;
X
Xfail:
X    free(fnbuffer);
X    return -1;
X}
X#endif
X
X
X#ifdef EMUL_OPEN3
X#include "open3.h"
X/*
X * open3 -- routine to emulate the 3-argument open system
X * call that is present in most modern Unix systems.
X * This version attempts to support all the flag bits except for O_NDELAY
X * and O_APPEND, which are silently ignored.  The emulation is not as efficient
X * as the real thing (at worst, 4 system calls instead of one), but there's
X * not much I can do about that.
X *
X * Written 6/10/87 by rmtodd@uokmax
X *
X * open3(path, flag, mode)
X * Attempts to open the file specified by
X * the given pathname.  The following flag bits (#defined in tar.h)
X * specify options to the routine:
X *  O_RDONLY  file open for read only
X *  O_WRONLY  file open for write only
X *  O_RDWR        file open for both read & write
X * (Needless to say, you should only specify one of the above).
X *  O_CREAT       file is created with specified mode if it needs to be.
X *  O_TRUNC       if file exists, it is truncated to 0 bytes
X *  O_EXCL        used with O_CREAT--routine returns error if file exists
X * Function returns file descriptor if successful, -1 and errno if not.
X */
X
X/*
X * array to give arguments to access for various modes
X * FIXME, this table depends on the specific integer values of O_XXX,
X * and also contains integers (args to 'access') that should be #define's.
X */
```

```
Xstatic int modes[] =
X    {
X         04, /* O_RDONLY */
X         02, /* O_WRONLY */
X         06, /* O_RDWR */
X         06, /* invalid but we'd better cope -- O_WRONLY+O_RDWR */
X    };
X
X/* Shut off the automatic emulation of open(), we'll need it. */
X#undef open
X
Xint
Xopen3(path, flags, mode)
Xchar *path;
Xint flags, mode;
X{
X    extern int errno;
X    int exists = 1;
X    int call_creat = 0;
X    int fd;
X    /*
X     * We actually do the work by calling the open() or creat() system
X     * call, depending on the flags.  Call_creat is true if we will use
X     * creat(), false if we will use open().
X     */
X
X    /*
X     * See if the file exists and is accessible in the requested mode.
X     *
X     * Strictly speaking we shouldn't be using access, since access checks
X     * against real uid, and the open call should check against euid.
X     * Most cases real uid == euid, so it won't matter.   FIXME.
X     * FIXME, the construction "flags & 3" and the modes table depends
X     * on the specific integer values of the O_XXX #define's.  Foo!
X     */
X    if (access(path,modes[flags & 3]) < 0) {
X         if (errno == ENOENT) {
X              /* the file does not exist */
X              exists = 0;
X         } else {
X              /* probably permission violation */
X              if (flags & O_EXCL) {
X                   /* Oops, the file exists, we didn't want it. */
X                   /* No matter what the error, claim EEXIST. */
X                   errno = EEXIST;
X              }
X              return -1;
X         }
X    }
X
X    /* if we have the O_CREAT bit set, check for O_EXCL */
X    if (flags & O_CREAT) {
X         if ((flags & O_EXCL) && exists) {
```

```
X             /* Oops, the file exists and we didn't want it to. */
X             errno = EEXIST;
X             return -1;
X         }
X         /*
X          * If the file doesn't exist, be sure to call creat() so that
X          * it will be created with the proper mode.
X          */
X         if (!exists) call_creat = 1;
X    } else {
X         /* If O_CREAT isn't set and the file doesn't exist, error. */
X         if (!exists) {
X             errno = ENOENT;
X             return -1;
X         }
X    }
X
X    /*
X     * If the O_TRUNC flag is set and the file exists, we want to call
X     * creat() anyway, since creat() guarantees that the file will be
X     * truncated and open()-for-writing doesn't.
X     * (If the file doesn't exist, we're calling creat() anyway and the
X     * file will be created with zero length.)
X     */
X    if ((flags & O_TRUNC) && exists) call_creat = 1;
X    /* actually do the call */
X    if (call_creat) {
X         /*
X          * call creat.  May have to close and reopen the file if we
X          * want O_RDONLY or O_RDWR access -- creat() only gives
X          * O_WRONLY.
X          */
X         fd = creat(path,mode);
X         if (fd < 0 || (flags & O_WRONLY)) return fd;
X         if (close(fd) < 0) return -1;
X         /* Fall out to reopen the file we've created */
X    }
X
X    /*
X     * calling old open, we strip most of the new flags just in case.
X     */
X    return open(path, flags & (O_RDONLY|O_WRONLY|O_RDWR|O_BINARY));
X}
X#endif
X
X#ifdef  WANT_MKNOD
X/* Fake mknod by complaining */
Xint
Xmknod(path, mode, dev)
X    char     *path;
X    unsigned short     mode;
X    dev_t         dev;
X{
```

```
X    extern int    errno;
X    int      fd;
X
X    errno = ENXIO;          /* No such device or address */
X    return -1;         /* Just give an error */
X}
X
X/* Fake links by copying */
Xint
Xlink(path1, path2)
X    char     *path1;
X    char     *path2;
X{
X    char buf[256];
X    int  ifd, ofd;
X    int  nrbytes;
X    int  nwbytes;
X
X    fprintf(stderr, "%s: %s: cannot link to %s, copying instead\n",
X         tar, path1, path2);
X    if ((ifd = open(path1, O_RDONLY|O_BINARY)) < 0)
X         return -1;
X    if ((ofd = creat(path2, 0666)) < 0)
X         return -1;
X    setmode(ofd, O_BINARY);
X    while ((nrbytes = read(ifd, buf, sizeof(buf))) > 0) {
X         if ((nwbytes = write(ofd, buf, nrbytes)) != nrbytes) {
X              nrbytes = -1;
X              break;
X         }
X    }
X    /* Note use of "|" rather than "||" below: we want to close
X     * the files even if an error occurs.
X     */
X    if ((nrbytes < 0) | (0 != close(ifd)) | (0 != close(ofd))) {
X         unlink(path2);
X         return -1;
X    }
X    return 0;
X}
X
X#ifndef atarist
X/* everyone owns everything on MS-DOS (or is it no one owns anything?) */
Xint
Xchown(path, uid, gid)
X    char *path;
X    int  uid;
X    int  gid;
X{
X    return 0;
X}
X
Xint
```

```
Xgeteuid()
X{
X    return 0;
X}
X#endif
X#endif  /* WANT_MKNOD */
X
X#ifdef WANT_UTILS
X/* Stash argv[0] here so panic will know what the program is called */
Xchar *myname = 0;
X
Xvoid
Xpanic(s)
Xchar *s;
X{
X    if(myname)
X         fprintf(stderr,"%s:",myname);
X    fprintf(stderr,s);
X    putc('\n',stderr);
X    exit(12);
X}
X
Xchar *
Xck_realloc(ptr,size)
Xchar *ptr;
Xsize_t size;
X{
X    char *ret;
X    char *realloc();
X
X    ret=realloc(ptr,size);
X    if(ret==0)
X         panic("Couldn't re-allocate memory");
X    return ret;
X}
X
X
X/* Implement a variable sized buffer of 'stuff'.  We don't know what it is,
X   nor do we care, as long as it doesn't mind being aligned on a char boundry.
X */
X
Xstruct buffer {
X    int  allocated;
X    int  length;
X    char *b;
X};
X
X#define MIN_ALLOCATE 50
X
Xchar *
Xinit_buffer()
X{
X    struct buffer *b;
```

```
X    char *ck_malloc();
X
X    b=(struct buffer *)ck_malloc(sizeof(struct buffer));
X    b->allocated=MIN_ALLOCATE;
X    b->b=(char *)ck_malloc(MIN_ALLOCATE);
X    b->length=0;
X    return (char *)b;
X}
X
Xvoid
Xflush_buffer(bb)
Xchar *bb;
X{
X    struct buffer *b;
X
X    b=(struct buffer *)bb;
X    free(b->b);
X    b->b=0;
X    b->allocated=0;
X    b->length=0;
X    free((void *)b);
X}
X
Xvoid
Xadd_buffer(bb,p,n)
Xchar *bb;
Xchar *p;
Xint n;
X{
X    struct buffer *b;
X
X    b=(struct buffer *)bb;
X    if(b->length+n>b->allocated) {
X         b->allocated*=2;
X         b->b=(char *)ck_realloc(b->b,b->allocated);
X    }
X    bcopy(p,b->b+b->length,n);
X    b->length+=n;
X}
X
Xchar *
Xget_buffer(bb)
Xchar *bb;
X{
X    struct buffer *b;
X
X    b=(struct buffer *)bb;
X    return b->b;
X}
X
Xchar *
Xmerge_sort(list,n,off,cmp)
Xchar *list;
```

```
Xint (*cmp)();
Xunsigned n;
Xint off;
X{
X    char *ret;
X
X    char *alist,*blist;
X    unsigned alength,blength;
X
X    char *tptr;
X    int tmp;
X    char **prev;
X#define NEXTOF(ptr)   (* ((char **)(((char *)(ptr))+off) ) )
X    if(n==1)
X         return list;
X    if(n==2) {
X         if((*cmp)(list,NEXTOF(list))>0) {
X              ret=NEXTOF(list);
X              NEXTOF(ret)=list;
X              NEXTOF(list)=0;
X              return ret;
X         }
X         return list;
X    }
X    alist=list;
X    alength=(n+1)/2;
X    blength=n/2;
X    for(tptr=list,tmp=(n-1)/2;tmp;tptr=NEXTOF(tptr),tmp--)
X         ;
X    blist=NEXTOF(tptr);
X    NEXTOF(tptr)=0;
X
X    alist=merge_sort(alist,alength,off,cmp);
X    blist=merge_sort(blist,blength,off,cmp);
X    prev = &ret;
X    for(;alist && blist;) {
X         if((*cmp)(alist,blist)<0) {
X              tptr=NEXTOF(alist);
X              *prev = alist;
X              prev = &(NEXTOF(alist));
X              alist=tptr;
X         } else {
X              tptr=NEXTOF(blist);
X              *prev = blist;
X              prev = &(NEXTOF(blist));
X              blist=tptr;
X         }
X    }
X    if(alist)
X         *prev = alist;
X    else
X         *prev = blist;
X
```

```
X    return ret;
X}
X
Xchar *
Xck_malloc(size)
Xsize_t size;
X{
X    char *ret;
X    char *malloc();
X
X    ret=malloc(size);
X    if(ret==0)
X         panic("Couldn't allocate memory");
X    return ret;
X}
X
Xvoid
Xck_close(fd)
Xint fd;
X{
X    if(close(fd)<0) {
X         msg_perror("can't close a file");
X         exit(EX_SYSTEM);
X    }
X}
X
X#include <ctype.h>
X#ifdef sparc
X#include "alloca.h"
X#endif
X
X/* Quote_copy_string is like quote_string, but instead of modifying the
X   string in place, it malloc-s a copy  of the string, and returns that.
X   If the string does not have to be quoted, it returns the NULL string.
X   The allocated copy can, of course, be freed with free() after the
X   caller is done with it.
X */
Xchar *
Xquote_copy_string(string)
Xchar *string;
X{
X    char *ret;
X    char *from_here;
X    char *to_there = 0;
X    char *copy_buf = 0;
X    int  c;
X    int  copying = 0;
X    void *alloca();
X
X    ret=string;
X
X    from_here=string;
X    while(*from_here) {
```

```
X        c= *from_here++;
X        if(c=='\\') {
X             if(!copying) {
X                  int n;
X
X                  n=(from_here-string)-1;
X                  copying++;
X                  copy_buf=(char *)malloc(n+1+strlen(from_here)*4);
X                  bcopy(string,copy_buf,n);
X                  to_there=copy_buf+n;
X             }
X             *to_there++='\\';
X             *to_there++='\\';
X        } else if(isprint(c)) {
X             if(copying)
X                  *to_there++= c;
X        } else {
X             if(!copying) {
X                  int  n;
X
X                  n=(from_here-string)-1;
X                  copying++;
X                  copy_buf=(char *)malloc(n+1+strlen(from_here)*4);
X                  bcopy(string,copy_buf,n);
X                  to_there=copy_buf+n;
X             }
X             *to_there++='\\';
X             if(c=='\n') *to_there++='n';
X             else if(c=='\t') *to_there++='t';
X             else if(c=='\f') *to_there++='f';
X             else if(c=='\b') *to_there++='b';
X             else if(c=='\r') *to_there++='r';
X             else if(c=='\177') *to_there++='?';
X             else {
X                  to_there[0]=(c>>6)+'0';
X                  to_there[1]=((c>>3)&07)+'0';
X                  to_there[2]=(c&07)+'0';
X                  to_there+=3;
X             }
X        }
X    }
X    if(copying) {
X         *to_there='\0';
X         return copy_buf;
X    }
X    return (char *)0;
X}
X
X
X/* Un_quote_string takes a quoted c-string (like those produced by
X   quote_string or quote_copy_string and turns it back into the
X   un-quoted original.  This is done in place.
X
```

```
X   HACK FOR ATARI ST: change '\' into '/', so people can use GEMDOS
X   style path names; this loses if un_quote_string gets called by
X   anyone but "name_next"!!!!!!!
X */
X
X/* There is no un-quote-copy-string.  Write it yourself */
X
Xchar *un_quote_string(string)
Xchar *string;
X{
X    char *ret;
X    char *from_here;
X    char *to_there;
X    int  tmp;
X
X    ret=string;
X    to_there=string;
X    from_here=string;
X    while(*from_here) {
X         if(*from_here!='\\') {
X              if(from_here!=to_there)
X                   *to_there++= *from_here++;
X              else
X                   from_here++,to_there++;
X              continue;
X         }
X#ifdef atarist
X         *to_there++ = '/';
X         from_here++;
X#else
X         switch(*++from_here) {
X         case '\\':
X              *to_there++= *from_here++;
X              break;
X         case 'n':
X              *to_there++= '\n';
X              from_here++;
X              break;
X         case 't':
X              *to_there++= '\t';
X              from_here++;
X              break;
X         case 'f':
X              *to_there++= '\f';
X              from_here++;
X              break;
X         case 'b':
X              *to_there++= '\b';
X              from_here++;
X              break;
X         case 'r':
X              *to_there++= '\r';
X              from_here++;
```

```
X             break;
X        case '?':
X             *to_there++= 0177;
X             from_here++;
X             break;
X        case '0':
X        case '1':
X        case '2':
X        case '3':
X        case '4':
X        case '5':
X        case '6':
X        case '7':
X             tmp= *from_here - '0';
X             from_here++;
X             if(*from_here<'0' || *from_here>'7') {
X                  *to_there++= tmp;
X                  break;
X             }
X             tmp= tmp*8 + *from_here-'0';
X             from_here++;
X             if(*from_here<'0' || *from_here>'7') {
X                  *to_there++= tmp;
X                  break;
X             }
X             tmp=tmp*8 + *from_here-'0';
X             from_here++;
X             *to_there=tmp;
X             break;
X        default:
X             ret=0;
X             *to_there++='\\';
X             *to_there++= *from_here++;
X             break;
X        }
X#endif
X    }
X    *to_there++='\0';
X    return ret;
X}
X#endif
X
X#ifdef WANT_CK_PIPE
Xvoid ck_pipe(pipes)
Xint *pipes;
X{
X    if(pipe(pipes)<0) {
X         msg_perror("can't open a pipe");
X         exit(EX_SYSTEM);
X    }
X}
X
Xchar *
```

```
Xgetwd(path)
Xchar *path;
X{
X    FILE *fp;
X    FILE *popen();
X
X    fp=popen("pwd","r");
X    if(!fp)
X         return 0;
X    if(!fgets(path,100,fp))
X         return 0;
X    if(!pclose(fp))
X         return 0;
X    return path;
X}
X#endif /* WANT_CK_PIPE */
X
X
X#ifdef WANT_GETOPT
X/* Getopt for GNU.
X   Copyright (C) 1987 Free Software Foundation, Inc.
X
X               NO WARRANTY
X
X  BECAUSE THIS PROGRAM IS LICENSED FREE OF CHARGE, WE PROVIDE ABSOLUTELY
XNO WARRANTY, TO THE EXTENT PERMITTED BY APPLICABLE STATE LAW.  EXCEPT
XWHEN OTHERWISE STATED IN WRITING, FREE SOFTWARE FOUNDATION, INC,
XRICHARD M. STALLMAN AND/OR OTHER PARTIES PROVIDE THIS PROGRAM "AS IS"
XWITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING,
XBUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND
XFITNESS FOR A PARTICULAR PURPOSE.  THE ENTIRE RISK AS TO THE QUALITY
XAND PERFORMANCE OF THE PROGRAM IS WITH YOU.  SHOULD THE PROGRAM PROVE
XDEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR
XCORRECTION.
X
X IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW WILL RICHARD M.
XSTALLMAN, THE FREE SOFTWARE FOUNDATION, INC., AND/OR ANY OTHER PARTY
XWHO MAY MODIFY AND REDISTRIBUTE THIS PROGRAM AS PERMITTED BELOW, BE
XLIABLE TO YOU FOR DAMAGES, INCLUDING ANY LOST PROFITS, LOST MONIES, OR
XOTHER SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE
XUSE OR INABILITY TO USE (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR
XDATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY THIRD PARTIES OR
XA FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS) THIS
XPROGRAM, EVEN IF YOU HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH
XDAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY.
X
X        GENERAL PUBLIC LICENSE TO COPY
X
X  1. You may copy and distribute verbatim copies of this source file
Xas you receive it, in any medium, provided that you conspicuously and
Xappropriately publish on each copy a valid copyright notice "Copyright
X (C) 1987 Free Software Foundation, Inc."; and include following the
Xcopyright notice a verbatim copy of the above disclaimer of warranty
```

Xand of this License. You may charge a distribution fee for the
Xphysical act of transferring a copy.
X
X 2. You may modify your copy or copies of this source file or
Xany portion of it, and copy and distribute such modifications under
Xthe terms of Paragraph 1 above, provided that you also do the following:
X
X   a) cause the modified files to carry prominent notices stating
X   that you changed the files and the date of any change; and
X
X   b) cause the whole of any work that you distribute or publish,
X   that in whole or in part contains or is a derivative of this
X   program or any part thereof, to be licensed at no charge to all
X   third parties on terms identical to those contained in this
X   License Agreement (except that you may choose to grant more
X   extensive warranty protection to third parties, at your option).
X
X   c) You may charge a distribution fee for the physical act of
X   transferring a copy, and you may at your option offer warranty
X   protection in exchange for a fee.
X
X 3. You may copy and distribute this program or any portion of it in
Xcompiled, executable or object code form under the terms of Paragraphs
X1 and 2 above provided that you do the following:
X
X   a) cause each such copy to be accompanied by the
X   corresponding machine-readable source code, which must
X   be distributed under the terms of Paragraphs 1 and 2 above; or,
X
X   b) cause each such copy to be accompanied by a
X   written offer, with no time limit, to give any third party
X   free (except for a nominal shipping charge) a machine readable
X   copy of the corresponding source code, to be distributed
X   under the terms of Paragraphs 1 and 2 above; or,
X
X   c) in the case of a recipient of this program in compiled, executable
X   or object code form (without the corresponding source code) you
X   shall cause copies you distribute to be accompanied by a copy
X   of the written offer of source code which you received along
X   with the copy you received.
X
X 4. You may not copy, sublicense, distribute or transfer this program
Xexcept as expressly provided under this License Agreement. Any attempt
Xotherwise to copy, sublicense, distribute or transfer this program is void and
Xyour rights to use the program under this License agreement shall be
Xautomatically terminated. However, parties who have received computer
Xsoftware programs from you with this License Agreement will not have
Xtheir licenses terminated so long as such parties remain in full compliance.
X
X 5. If you wish to incorporate parts of this program into other free
Xprograms whose distribution conditions are different, write to the Free
XSoftware Foundation at 675 Mass Ave, Cambridge, MA 02139. We have not yet
Xworked out a simple rule that can be stated here, but we will often permit

```
Xthis.  We will be guided by the two goals of preserving the free status of
Xall derivatives of our free software and of promoting the sharing and reuse of
Xsoftware.
X
X
XIn other words, you are welcome to use, share and improve this program.
XYou are forbidden to forbid anyone else to use, share and improve
Xwhat you give them.   Help stamp out software-hoarding!  */
X

X/* This version of `getopt' appears to the caller like standard Unix `getopt'
X   but it behaves differently for the user, since it allows the user
X   to intersperse the options with the other arguments.
X
X   As `getopt' works, it permutes the elements of `argv' so that,
X   when it is done, all the options precede everything else.  Thus
X   all application programs are extended to handle flexible argument order.
X
X   Setting the environment variable _POSIX_OPTION_ORDER disables permutation.
X   Then the behavior is completely standard.
X
X   GNU application programs can use a third alternative mode in which
X   they can distinguish the relative order of options and other arguments.  */
X
X#include <stdio.h>
X
X/* For communication from `getopt' to the caller.
X   When `getopt' finds an option that takes an argument,
X   the argument value is returned here.
X   Also, when `ordering' is RETURN_IN_ORDER,
X   each non-option ARGV-element is returned here.  */
X
Xchar *optarg = 0;
X
X/* Index in ARGV of the next element to be scanned.
X   This is used for communication to and from the caller
X   and for communication between successive calls to `getopt'.
X
X   On entry to `getopt', zero means this is the first call; initialize.
X
X   When `getopt' returns EOF, this is the index of the first of the
X   non-option elements that the caller should itself scan.
X
X   Otherwise, `optind' communicates from one call to the next
X   how much of ARGV has been scanned so far.  */
X
Xint optind = 0;
X
X/* The next char to be scanned in the option-element
X   in which the last option character we returned was found.
X   This allows us to pick up the scan where we left off.
X
X   If this is zero, or a null string, it means resume the scan
```

```
X   by advancing to the next ARGV-element.  */
X
Xstatic char *nextchar;
X
X/* Callers store zero here to inhibit the error message
X   for unrecognized options.  */
X
Xint opterr = 1;
X
X/* Describe how to deal with options that follow non-option ARGV-elements.
X
X   UNSPECIFIED means the caller did not specify anything;
X   the default is then REQUIRE_ORDER if the environment variable
X   _OPTIONS_FIRST is defined, PERMUTE otherwise.
X
X   REQUIRE_ORDER means don't recognize them as options.
X   Stop option processing when the first non-option is seen.
X   This is what Unix does.
X
X   PERMUTE is the default.  We permute the contents of `argv' as we scan,
X   so that eventually all the options are at the end.  This allows options
X   to be given in any order, even with programs that were not written to
X   expect this.
X
X   RETURN_IN_ORDER is an option available to programs that were written
X   to expect options and other ARGV-elements in any order and that care about
X   the ordering of the two.  We describe each non-option ARGV-element
X   as if it were the argument of an option with character code zero.
X   Using `-' as the first character of the list of option characters
X   requests this mode of operation.
X
X   The special argument `--' forces an end of option-scanning regardless
X   of the value of `ordering'.  In the case of RETURN_IN_ORDER, only
X   `--' can cause `getopt' to return EOF with `optind' != ARGC.  */
X
Xstatic enum { REQUIRE_ORDER, PERMUTE, RETURN_IN_ORDER } ordering;
X

X/* Handle permutation of arguments.  */
X
X/* Describe the part of ARGV that contains non-options that have
X   been skipped.  `first_nonopt' is the index in ARGV of the first of them;
X   `last_nonopt' is the index after the last of them.  */
X
Xstatic int first_nonopt;
Xstatic int last_nonopt;
X
X/* Exchange two adjacent subsequences of ARGV.
X   One subsequence is elements [first_nonopt,last_nonopt)
X    which contains all the non-options that have been skipped so far.
X   The other is elements [last_nonopt,optind), which contains all
X    the options processed since those non-options were skipped.
X
```

```
X   `first_nonopt' and `last_nonopt' are relocated so that they describe
X    the new indices of the non-options in ARGV after they are moved.  */
X
Xstatic void
Xexchange (argv)
X     char **argv;
X{
X  int nonopts_size
X    = (last_nonopt - first_nonopt) * sizeof (char *);
X  char **temp = (char **) alloca (nonopts_size);
X
X  /* Interchange the two blocks of data in argv.  */
X
X  bcopy (&argv[first_nonopt], temp, nonopts_size);
X  bcopy (&argv[last_nonopt], &argv[first_nonopt],
X     (optind - last_nonopt) * sizeof (char *));
X  bcopy (temp, &argv[first_nonopt + optind - last_nonopt],
X     nonopts_size);
X
X  /* Update records for the slots the non-options now occupy.  */
X
X  first_nonopt += (optind - last_nonopt);
X  last_nonopt = optind;
X}
X

X/* Scan elements of ARGV (whose length is ARGC) for option characters
X   given in OPTSTRING.
X
X   If an element of ARGV starts with '-', and is not exactly "-" or "--",
X   then it is an option element.  The characters of this element
X   (aside from the initial '-') are option characters.  If `getopt'
X   is called repeatedly, it returns successively each of theoption characters
X   from each of the option elements.
X
X   If `getopt' finds another option character, it returns that character,
X   updating `optind' and `nextchar' so that the next call to `getopt' can
X   resume the scan with the following option character or ARGV-element.
X
X   If there are no more option characters, `getopt' returns `EOF'.
X   Then `optind' is the index in ARGV of the first ARGV-element
X   that is not an option.  (The ARGV-elements have been permuted
X   so that those that are not options now come last.)
X
X   OPTSTRING is a string containing the legitimate option characters.
X   A colon in OPTSTRING means that the previous character is an option
X   that wants an argument.  The argument is taken from the rest of the
X   current ARGV-element, or from the following ARGV-element,
X   and returned in `optarg'.
X
X   If an option character is seen that is not listed in OPTSTRING,
X   return '?' after printing an error message.  If you set `opterr' to
X   zero, the error message is suppressed but we still return '?'.
```

```
X
X   If a char in OPTSTRING is followed by a colon, that means it wants an arg,
X   so the following text in the same ARGV-element, or the text of the following
X   ARGV-element, is returned in `optarg.  Two colons mean an option that
X   wants an optional arg; if there is text in the current ARGV-element,
X   it is returned in `optarg'.
X
X   If OPTSTRING starts with `-', it requests a different method of handling the
X   non-option ARGV-elements.  See the comments about RETURN_IN_ORDER, above.  */
X
Xint
Xgetopt (argc, argv, optstring)
X     int argc;
X     char **argv;
X     char *optstring;
X{
X  /* Initialize the internal data when the first call is made.
X     Start processing options with ARGV-element 1 (since ARGV-element 0
X     is the program name); the sequence of previously skipped
X     non-option ARGV-elements is empty.  */
X
X  if (optind == 0)
X    {
X      first_nonopt = last_nonopt = optind = 1;
X
X      nextchar = 0;
X
X      /* Determine how to handle the ordering of options and nonoptions.  */
X
X      if (optstring[0] == '-')
X    ordering = RETURN_IN_ORDER;
X      else if (getenv ("_POSIX_OPTION_ORDER") != 0)
X    ordering = REQUIRE_ORDER;
X      else
X    ordering = PERMUTE;
X    }
X
X  if (nextchar == 0 || *nextchar == 0)
X    {
X      if (ordering == PERMUTE)
X    {
X      /* If we have just processed some options following some non-options,
X         exchange them so that the options come first.  */
X
X      if (first_nonopt != last_nonopt && last_nonopt != optind)
X        exchange (argv);
X      else if (last_nonopt != optind)
X        first_nonopt = optind;
X
X      /* Now skip any additional non-options
X         and extend the range of non-options previously skipped.  */
X
X      while (optind < argc
```

```
X         && (argv[optind][0] != '-'
X             || argv[optind][1] == 0))
SHAR_EOF
echo "End of part 4"
echo "File PORT.C is continued in part 5"
echo "5" > s2_seq_.tmp
exit 0


From -438250767536992695
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,3bc195fae4277b7b,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-19 14:43:23 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
agate!apple!sun-barr!newstop!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i051: gnutar -- GNU's version of UNIX "tar"  part05/08
Keywords: shar, C
Message-ID: <33891@grapevine.uucp>
Date: 19 Jul 89 21:43:23 GMT
Sender: news@grapevine.uucp
Lines: 1877
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: 7103_300@uwovax.uwo.ca (Eric R. Smith)
Posting-number: Volume 2, Issue 51
Archive-name: gnutar/part05

#!/bin/sh
# this is part 5 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file PORT.C continued
#
CurArch=5
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file PORT.C"
sed 's/^X//' << 'SHAR_EOF' >> PORT.C
X       optind++;
X     last_nonopt = optind;
X   }
X
X     /* Special ARGV-element `--' means premature end of options.
X    Skip it like a null option,
```

```
X     then exchange with previous non-options as if it were an option,
X     then skip everything else like a non-option.  */
X
X      if (optind != argc && !strcmp (argv[optind], "--"))
X    {
X      optind++;
X
X      if (first_nonopt != last_nonopt && last_nonopt != optind)
X        exchange (argv);
X      else if (first_nonopt == last_nonopt)
X        first_nonopt = optind;
X      last_nonopt = argc;
X
X      optind = argc;
X    }
X
X      /* If we have done all the ARGV-elements, stop the scan
X     and back over any non-options that we skipped and permuted.  */
X
X      if (optind == argc)
X    {
X      /* Set the next-arg-index to point at the non-options
X         that we previously skipped, so the caller will digest them.  */
X      if (first_nonopt != last_nonopt)
X        optind = first_nonopt;
X      return EOF;
X    }
X
X      /* If we have come to a non-option and did not permute it,
X     either stop the scan or describe it to the caller and pass it by.  */
X
X      if (argv[optind][0] != '-' || argv[optind][1] == 0)
X    {
X      if (ordering == REQUIRE_ORDER)
X        return EOF;
X      optarg = argv[optind++];
X      return 0;
X    }
X
X      /* We have found another option-ARGV-element.
X     Start decoding its characters.  */
X
X      nextchar = argv[optind] + 1;
X    }
X
X  /* Look at and handle the next option-character.  */
X
X  {
X    char c = *nextchar++;
X    char *temp = (char *) index (optstring, c);
X
X    /* Increment `optind' when we start to process its last character.  */
X    if (*nextchar == 0)
```

```
X      optind++;
X
X    if (temp == 0)
X      {
X    if (opterr != 0)
X      {
X        if (c < 040 || c >= 0177)
X          fprintf (stderr, "%s: unrecognized option, character code 0%o\n",
X                argv[0], c);
X        else
X          fprintf (stderr, "%s: unrecognized option `-%c'\n",
X                argv[0], c);
X      }
X    return '?';
X      }
X    if (temp[1] == ':')
X      {
X    if (temp[2] == ':')
X      {
X        /* This is an option that accepts an argument optionally.  */
X        if (*nextchar != 0)
X          {
X            optarg = nextchar;
X         optind++;
X          }
X        else
X          optarg = 0;
X        nextchar = 0;
X      }
X    else
X      {
X        /* This is an option that requires an argument.  */
X        if (*nextchar != 0)
X          {
X         optarg = nextchar;
X         /* If we end this ARGV-element by taking the rest as an arg,
X            we must advance to the next element now.  */
X         optind++;
X          }
X        else if (optind == argc)
X          {
X         if (opterr != 0)
X           fprintf (stderr, "%s: no argument for `-%c' option\n",
X                 argv[0], c);
X         optarg = 0;
X          }
X        else
X          /* We already incremented `optind' once;
X           increment it again when taking next ARGV-elt as argument.  */
X          optarg = argv[optind++];
X        nextchar = 0;
X      }
X      }
```

```
X    return c;
X  }
X}
X

X#ifdef TEST
X
X/* Compile with -DTEST to make an executable for use in testing
X   the above definition of `getopt'.  */
X
Xint
Xmain (argc, argv)
X     int argc;
X     char **argv;
X{
X  char c;
X  int digit_optind = 0;
X
X  while (1)
X    {
X      int this_option_optind = optind;
X      if ((c = getopt (argc, argv, "abc:d:0123456789")) == EOF)
X    break;
X
X      switch (c)
X    {
X    case '0':
X    case '1':
X    case '2':
X    case '3':
X    case '4':
X    case '5':
X    case '6':
X    case '7':
X    case '8':
X    case '9':
X      if (digit_optind != 0 && digit_optind != this_option_optind)
X        printf ("digits occur in two different argv-elements.\n");
X      digit_optind = this_option_optind;
X      printf ("option %c\n", c);
X      break;
X
X    case 'a':
X      printf ("option a\n");
X      break;
X
X    case 'b':
X      printf ("option b\n");
X      break;
X
X    case 'c':
X      printf ("option c with value `%s'\n", optarg);
X      break;
```

```
X
X    case '?':
X      break;
X
X    default:
X      printf ("?? getopt returned character code 0%o ??\n", c);
X    }
X    }
X
X  if (optind < argc)
X    {
X      printf ("non-option ARGV-elements: ");
X      while (optind < argc)
X    printf ("%s ", argv[optind++]);
X      printf ("\n");
X    }
X
X  return 0;
X}
X
X#endif /* TEST */
X
X#endif /* WANT_GETOPT */
X
X#ifdef WANT_STRSTR
X
X/*
X * strstr - find first occurrence of wanted in s
X */
X
Xchar *                /* found string, or NULL if none */
Xstrstr(s, wanted)
Xchar *s;
Xchar *wanted;
X{
X    register char *scan;
X    register size_t len;
X    register char firstc;
X    extern int strcmp();
X
X    /*
X     * The odd placement of the two tests is so "" is findable.
X     * Also, we inline the first char for speed.
X     * The ++ on scan has been moved down for optimization.
X     */
X    firstc = *wanted;
X    len = strlen(wanted);
X    for (scan = s; *scan != firstc || strncmp(scan, wanted, len) != 0; )
X        if (*scan++ == '\0')
X            return (char *)0;
X    return scan;
X}
X#endif
```

```
X
X#ifdef STDC_MSG
X#include <stdarg.h>
X
X
Xvoid
Xmsg(char *str,...)
X{
X    va_list args;
X
X    va_start(args,str);
X    fprintf(stderr,"%s: ",tar);
X    if(f_sayblock)
X         fprintf(stderr,"rec %d: ",baserec + (ar_record - ar_block));
X    vfprintf(stderr,str,args);
X    va_end(args);
X    putc('\n',stderr);
X}
X
Xvoid
Xmsg_perror(char *str,...)
X{
X    va_list args;
X    int save_e;
X    extern int errno;
X
X    save_e=errno;
X    fprintf(stderr,"%s: ",tar);
X    if(f_sayblock)
X         fprintf(stderr,"rec %d: ",baserec + (ar_record - ar_block));
X    va_start(args,str);
X    vfprintf(stderr,str,args);
X    va_end(args);
X    fprintf(stderr,": ");
X    errno=save_e;
X    perror((char *)0);
X}
X#endif
X
X#ifdef VARARGS_MSG
X#include <varargs.h>
Xvoid msg(str,va_alist)
Xchar *str;
Xva_dcl
X{
X    va_list args;
X
X    fprintf(stderr,"%s: ",tar);
X    if(f_sayblock)
X         fprintf(stderr,"rec %d: ",baserec + (ar_record - ar_block));
X    va_start(args);
X    vfprintf(stderr,str,args);
X    va_end(args);
```

```
X    putc('\n',stderr);
X}
X
Xvoid msg_perror(str,va_alist)
Xchar *str;
Xva_dcl
X{
X    va_list args;
X    int save_e;
X    extern int errno;
X
X    save_e=errno;
X    fprintf(stderr,"%s: ",tar);
X    if(f_sayblock)
X         fprintf(stderr,"rec %d: ",baserec + (ar_record - ar_block));
X    va_start(args);
X    vfprintf(stderr,str,args);
X    va_end(args);
X    fprintf(stderr,": ");
X    errno=save_e;
X    perror(0);
X}
X#endif
X
X#ifdef DOPRNT_MSG
Xvoid msg(str,args)
Xchar *str;
X{
X    fprintf(stderr,"%s: ",tar);
X    if(f_sayblock)
X         fprintf(stderr,"rec %d: ",baserec + (ar_record - ar_block));
X    _doprnt(str, &args, stderr);
X    putc('\n',stderr);
X}
X
Xvoid msg_perror(str,args)
Xchar *str;
X{
X    int save_e;
X    extern int errno;
X
X    save_e=errno;
X    fprintf(stderr,"%s: ",tar);
X    if(f_sayblock)
X         fprintf(stderr,"rec %d: ",baserec + (ar_record - ar_block));
X    _doprnt(str, &args, stderr);
X    fprintf(stderr,": ");
X    errno=save_e;
X    perror(0);
X}
X
X#endif
X#ifdef LOSING_MSG
```

```
Xvoid msg(str,a1,a2,a3,a4,a5,a6)
Xchar *str;
X{
X    fprintf(stderr,"%s: ",tar);
X    if(f_sayblock)
X        fprintf(stderr,"rec %d: ",baserec + (ar_record - ar_block));
X    fprintf(stderr,str,a1,a2,a3,a4,a5,a6);
X    putc('\n',stderr);
X}
X
Xvoid msg_perror(str,a1,a2,a3,a4,a5,a6)
Xchar *str;
X{
X    int save_e;
X    extern int errno;
X
X    save_e=errno;
X    fprintf(stderr,"%s: ",tar);
X    if(f_sayblock)
X        fprintf(stderr,"rec %d: ",baserec + (ar_record - ar_block));
X    fprintf(stderr,str,a1,a2,a3,a4,a5,a6);
X    fprintf(stderr,": ");
X    errno=save_e;
X    perror(0);
X}
X
X#endif
SHAR_EOF
echo "File PORT.C is complete"
chmod 0644 PORT.C || echo "restore of PORT.C fails"
echo "x - extracting PORT.H (Text)"
sed 's/^X//' << 'SHAR_EOF' > PORT.H &&
X/*
X
X    Copyright (C) 1988 Free Software Foundation
X
XGNU tar is distributed in the hope that it will be useful, but WITHOUT ANY
XWARRANTY.  No author or distributor accepts responsibility to anyone
Xfor the consequences of using it or for whether it serves any
Xparticular purpose or works at all, unless he says so in writing.
XRefer to the GNU tar General Public License for full details.
X
XEveryone is granted permission to copy, modify and redistribute GNU tar,
Xbut only under the conditions described in the GNU tar General Public
XLicense.  A copy of this license is supposed to have been given to you
Xalong with GNU tar so you can know your rights and responsibilities.  It
Xshould be in a file named COPYING.  Among other things, the copyright
Xnotice and this notice must be preserved on all copies.
X
XIn other words, go ahead and share GNU tar, but don't try to stop
Xanyone else from sharing it farther.  Help stamp out software hoarding!
X*/
X/*
```

```
X * Portability declarations for tar.
X *
X * @(#)port.h 1.3 87/11/11  by John Gilmore, 1986
X */
X
X/*
X * Everybody does wait() differently.  There seem to be no definitions
X * for this in V7 (e.g. you are supposed to shift and mask things out
X * using constant shifts and masks.)  So fuck 'em all -- my own non
X * standard but portable macros.  Don't change to a "union wait"
X * based approach -- the ordering of the elements of the struct
X * depends on the byte-sex of the machine.  Foo!
X */
X#define  TERM_SIGNAL(status)    ((status) & 0x7F)
X#define TERM_COREDUMP(status)   (((status) & 0x80) != 0)
X#define TERM_VALUE(status) ((status) >> 8)
X
X#ifdef  MSDOS
X/* missing things from sys/stat.h */
X#ifndef atarist
X#define  S_ISUID       0
X#define  S_ISGID       0
X#endif
X#define  S_ISVTX       0
X
X/* device stuff */
X#define  makedev(ma, mi)        ((ma << 8) | mi)
X#define  major(dev)         (dev)
X#define  minor(dev)         (dev)
X#endif  /* MSDOS */
X
X#ifdef atarist
X#define MAXNAMLEN 64
X#endif
SHAR_EOF
chmod 0644 PORT.H || echo "restore of PORT.H fails"
echo "x - extracting README (Text)"
sed 's/^X//' << 'SHAR_EOF' > README &&
XThis is GNU tar.  It is based heavily on John Gilmore's public domain tar,
Xbut with added features.  See tar.texinfo for details.
X
XPlease send bug reports, etc to hack@gnu.ai.mit.edu hack@wheaties.ai.mit.edu
Xor hack@media-lab.media.mit.edu
X
X             hack
SHAR_EOF
chmod 0644 README || echo "restore of README fails"
echo "x - extracting README.ST (Text)"
sed 's/^X//' << 'SHAR_EOF' > README.ST &&
XHere is GNU tar, ported to the Atari ST. The source code is included
X(of course!). See the file "copying" for copyright information.
XPLEASE NOTE: THERE IS ABSOLUTELY NO WARRANTY FOR THIS PRODUCT. USE IT
XAT YOUR OWN RISK!
```

```
X
XAll the changes to the source that were necessary for this version
Xare surrounded by "#ifdef atarist". I used the GCC v1.34, with a new library,
Xto build it. If you're using another compiler, you may need to do some
Xwork. In particular, you'll probably want to add -Datarist to the CFLAGS
Xin the makefile.
X
XRead the enclosed manual (in tar.tex -- actually this should be
Xtar.texinfo, but GEMDOS doesn't know from long extensions) for general
Xinformation.
X
XST-specific information:
X
XTar is designed to run from a command line interpreter,
Xpreferably one that uses the MWC extended argument passing conventions
X(e.g. gulam). You can run it from the desktop if (a) your version of the
Xdesktop will pass lower case letters to programs, and (b) you don't mind
Xmessages disappearing from the screen when the desktop gets redrawn.
X
XThere is no default tar file; one must be explicitly specified on the
Xcommand line. The "-f" option is assumed, i.e typing "tar tv junk.tar"
Xis the same as "tar tvf junk.tar".
X
XSeveral of the options (e.g. -z to compress) are not applicable on the
XST, and will generate error messages.
X
XThe -N option only accepts a date in the form MM/DD/YYYY. (Theoretically,
Xif you have a shell that uses the MWC extended argument passing conventions,
Xyou could use "-NMM/DD/YYY hh:mm:ss" -- make sure you put quotes around
Xit).
X
XTimezones: generally, you will want to tell the program which time zone
Xyou're currently in. To do this, create an environment variable
Xcalled TZ with the format SSS[-]nn[:nn][DDD], (things in [] optional) where
XSSS is the name of your time zone, nn:nn is the offset in hours and minutes
Xfrom GMT (prefixed by '-' if necessary) and DDD, if present, is the name
Xused for your time zone when daylight savings time is in effect. Note
Xthat the rules for daylight savings time are very primitive indeed,
Xand will probably not work outside of North America.
XExample: I'm in the N. American Eastern time zone, and I use (in Gulam)
Xsetenv TZ "EST5EDT".
XIf no TZ environment variable is found, GMT is assumed. This only matters
Xif you're transferring tar files to/from other systems and you want
Xto make sure the dates are correct.
X
XGEMDOS filenames are translated to/from Unix; thus "subdir/file" gets
Xchanged to "subdir\file". This can (obviously) cause problems with GEMDOS
Xfiles which contain '/' in their names, or Unix files that contain '\'.
XWatch out for these. Also, some effort is made to translate names that
Xhave several extensions, e.g. "junk.c.bak" gets mapped to "junk,c.bak".
X
XMulti-volume tar files seem to work, but have not been thoroughly tested.
XI would not recommend using them to back up your hard disk, unless you're
```

```
Xfeeling particularly brave or don't care about your data.
X
XReport bugs, etc. to me at the address below:
X
XEric R. Smith                      email:
XDept. of Mathematics            7103_300@uwovax.uwo.ca
XUniversity of Western Ontario   7103_300@uwovax.bitnet
XLondon, Ont. Canada N6A 5B7     (a shared mailbox: put my name on
Xph: (519) 661-3638              the message, please!)
X
SHAR_EOF
chmod 0644 README.ST || echo "restore of README.ST fails"
echo "x - extracting RMT.H (Text)"
sed 's/^X//' << 'SHAR_EOF' > RMT.H &&
X/*
X
X    Copyright (C) 1988 Free Software Foundation
X
XGNU tar is distributed in the hope that it will be useful, but WITHOUT ANY
XWARRANTY.  No author or distributor accepts responsibility to anyone
Xfor the consequences of using it or for whether it serves any
Xparticular purpose or works at all, unless he says so in writing.
XRefer to the GNU tar General Public License for full details.
X
XEveryone is granted permission to copy, modify and redistribute GNU tar,
Xbut only under the conditions described in the GNU tar General Public
XLicense.  A copy of this license is supposed to have been given to you
Xalong with GNU tar so you can know your rights and responsibilities.  It
Xshould be in a file named COPYING.  Among other things, the copyright
Xnotice and this notice must be preserved on all copies.
X
XIn other words, go ahead and share GNU tar, but don't try to stop
Xanyone else from sharing it farther.  Help stamp out software hoarding!
X*/
X
X#ifdef NO_REMOTE
X#define rmtopen       open
X#define rmtaccess access
X#define rmtstat       stat
X#define rmtcreat  creat
X#define rmtlstat  lstat
X#define rmtread       read
X#define rmtwrite  write
X#define rmtlseek  lseek
X#define rmtclose  close
X#define rmtioctl  ioctl
X#define rmtdup        dup
X#define rmtfstat  fstat
X#define rmtfcntl  fcntl
X#define rmtisatty isatty
X
X#else
X#ifndef USG
```

```
X#define strchr index
X#endif
X
X#define __REM_BIAS    128
X#define RMTIOCTL
X
X#ifndef O_CREAT
X#define O_CREAT   01000
X#endif
Xextern char *__rmt_path;
Xextern char *strchr();
X
X#define _remdev(path) ((__rmt_path=strchr(path, ':')) && strncmp(__rmt_path, ":/
dev/", 6)==0)
X#define _isrmt(fd)         ((fd) >= __REM_BIAS)
X
X#define rmtopen(path,oflag,mode) (_remdev(path) ? __rmt_open(path, oflag, mode,
__REM_BIAS) : open(path, oflag, mode))
X#define rmtaccess(path, amode)  (_remdev(path) ? 0 : access(path, amode))
X#define rmtstat(path, buf) (_remdev(path) ? (errno = EOPNOTSUPP), -1 : stat(path,
buf))
X#define rmtcreat(path, mode)    (_remdev(path) ? __rmt_open (path, 1 | O_CREAT,
mode, __REM_BIAS) : creat(path, mode))
X#define rmtlstat(path,buf) (_remdev(path) ? (errno = EOPNOTSUPP), -1 :
lstat(path,buf))
X
X#define rmtread(fd, buf, n)     (_isrmt(fd) ? __rmt_read(fd - __REM_BIAS, buf,
n) : read(fd, buf, n))
X#define rmtwrite(fd, buf, n)    (_isrmt(fd) ? __rmt_write(fd - __REM_BIAS, buf,
n) : write(fd, buf, n))
X#define rmtlseek(fd, off, wh)   (_isrmt(fd) ? __rmt_lseek(fd - __REM_BIAS, off,
wh) : lseek(fd, off, wh))
X#define rmtclose(fd)       (_isrmt(fd) ? __rmt_close(fd - __REM_BIAS) :
close(fd))
X#ifdef RMTIOCTL
X#define rmtioctl(fd,req,arg)    (_isrmt(fd) ? __rmt_ioctl(fd - __REM_BIAS, req,
arg) : ioctl(fd, req, arg))
X#else
X#define rmtioctl(fd,req,arg)    (_isrmt(fd) ? (errno = EOPNOTSUPP), -1 :
ioctl(fd, req, arg))
X#endif
X#define rmtdup(fd)         (_isrmt(fd) ? (errno = EOPNOTSUPP), -1 : dup(fd))
X#define rmtfstat(fd, buf)  (_isrmt(fd) ? (errno = EOPNOTSUPP), -1 : fstat(fd,
buf))
X#define rmtfcntl(fd,cmd,arg)    (_isrmt(fd) ? (errno = EOPNOTSUPP), -1 : fcntl
(fd, cmd, arg))
X#define rmtisatty(fd)      (_isrmt(fd) ? 0 : isatty(fd))
X
X#undef RMTIOCTL
X#endif
SHAR_EOF
chmod 0644 RMT.H || echo "restore of RMT.H fails"
echo "x - extracting RTAPE_LI.C (Text)"
```

```
sed 's/^X//' << 'SHAR_EOF' > RTAPE_LI.C &&
X/*
X
X    Copyright (C) 1988 Free Software Foundation
X
XGNU tar is distributed in the hope that it will be useful, but WITHOUT ANY
XWARRANTY.  No author or distributor accepts responsibility to anyone
Xfor the consequences of using it or for whether it serves any
Xparticular purpose or works at all, unless he says so in writing.
XRefer to the GNU tar General Public License for full details.
X
XEveryone is granted permission to copy, modify and redistribute GNU tar,
Xbut only under the conditions described in the GNU tar General Public
XLicense.  A copy of this license is supposed to have been given to you
Xalong with GNU tar so you can know your rights and responsibilities.  It
Xshould be in a file named COPYING.  Among other things, the copyright
Xnotice and this notice must be preserved on all copies.
X
XIn other words, go ahead and share GNU tar, but don't try to stop
Xanyone else from sharing it farther.  Help stamp out software hoarding!
X*/
X
X#ifndef lint
Xstatic char *RCSid = "$Header: rmtlib.c,v 1.6 88/10/25 17:04:29 root Locked $";
X#endif
X
X/* JF: modified to make all rmtXXX calls into macros for speed */
X/*
X * $Log: rmtlib.c,v $
X * Revision 1.6  88/10/25  17:04:29  root
X * rexec code and a bug fix from srs!dan, miscellanious cleanup. ADR.
X *
X * Revision 1.5  88/10/25  16:30:17  root
X * Fix from jeff@gatech.edu for getting user@host:dev right. ADR.
X *
X * Revision 1.4  87/10/30  10:36:12  root
X * Made 4.2 syntax a compile time option. ADR.
X *
X * Revision 1.3  87/04/22  11:16:48  root
X * Two fixes from parmelee@wayback.cs.cornell.edu to correctly
X * do fd biasing and rmt protocol on 'S' command. ADR.
X *
X * Revision 1.2  86/10/09  16:38:53  root
X * Changed to reflect 4.3BSD rcp syntax. ADR.
X *
X * Revision 1.1  86/10/09  16:17:35  root
X * Initial revision
X *
X */
X
X/*
X *  rmt --- remote tape emulator subroutines
X *
```

```
X *  Originally written by Jeff Lee, modified some by Arnold Robbins
X *
X *  WARNING:  The man page rmt(8) for /etc/rmt documents the remote mag
X *  tape protocol which rdump and rrestore use.  Unfortunately, the man
X *  page is *WRONG*.  The author of the routines I'm including originally
X *  wrote his code just based on the man page, and it didn't work, so he
X *  went to the rdump source to figure out why.  The only thing he had to
X *  change was to check for the 'F' return code in addition to the 'E',
X *  and to separate the various arguments with \n instead of a space.  I
X *  personally don't think that this is much of a problem, but I wanted to
X *  point it out.
X *  -- Arnold Robbins
X *
X *  Redone as a library that can replace open, read, write, etc, by
X *  Fred Fish, with some additional work by Arnold Robbins.
X */
X
X/*
X *  MAXUNIT --- Maximum number of remote tape file units
X *
X *  READ --- Return the number of the read side file descriptor
X *  WRITE --- Return the number of the write side file descriptor
X */
X
X#define RMTIOCTL  1
X/* #define USE_REXEC  1    /* rexec code courtesy of Dan Kegel, srs!dan */
X
X#include <stdio.h>
X#include <signal.h>
X#include <sys/types.h>
X
X#ifdef RMTIOCTL
X#include <sys/ioctl.h>
X#include <sys/mtio.h>
X#endif
X
X#ifdef USE_REXEC
X#include <netdb.h>
X#endif
X
X#include <errno.h>
X#include <setjmp.h>
X#include <sys/stat.h>
X
X#define BUFMAGIC  64   /* a magic number for buffer sizes */
X#define MAXUNIT   4
X
X#define READ(fd)  (Ctp[fd][0])
X#define WRITE(fd) (Ptc[fd][1])
X
Xstatic int Ctp[MAXUNIT][2] = { -1, -1, -1, -1, -1, -1, -1, -1 };
Xstatic int Ptc[MAXUNIT][2] = { -1, -1, -1, -1, -1, -1, -1, -1 };
X
```

```
Xextern int errno;
X
Xchar *__rmt_path;
X
X/*
X *  _rmt_panic --- close off a remote tape connection
X */
X
Xstatic void _rmt_panic(fildes)
Xint fildes;
X{
X    close(READ(fildes));
X    close(WRITE(fildes));
X    READ(fildes) = -1;
X    WRITE(fildes) = -1;
X}
X
X
X
X/*
X *  command --- attempt to perform a remote tape command
X */
X
Xstatic int command(fildes, buf)
Xint fildes;
Xchar *buf;
X{
X    register int blen;
X    int (*pstat)();
X
X/*
X *  save current pipe status and try to make the request
X */
X
X    blen = strlen(buf);
X    pstat = signal(SIGPIPE, SIG_IGN);
X    if (write(WRITE(fildes), buf, blen) == blen)
X    {
X         signal(SIGPIPE, pstat);
X         return(0);
X    }
X
X/*
X *  something went wrong. close down and go home
X */
X
X    signal(SIGPIPE, pstat);
X    _rmt_panic(fildes);
X
X    errno = EIO;
X    return(-1);
X}
X
```

```
X
X
X/*
X * status --- retrieve the status from the pipe
X */
X
Xstatic int status(fildes)
Xint fildes;
X{
X    int i;
X    char c, *cp;
X    char buffer[BUFMAGIC];
X
X/*
X * read the reply command line
X */
X
X    for (i = 0, cp = buffer; i < BUFMAGIC; i++, cp++)
X    {
X         if (read(READ(fildes), cp, 1) != 1)
X         {
X              _rmt_panic(fildes);
X              errno = EIO;
X              return(-1);
X         }
X         if (*cp == '\n')
X         {
X              *cp = 0;
X              break;
X         }
X    }
X
X    if (i == BUFMAGIC)
X    {
X         _rmt_panic(fildes);
X         errno = EIO;
X         return(-1);
X    }
X
X/*
X * check the return status
X */
X
X    for (cp = buffer; *cp; cp++)
X         if (*cp != ' ')
X              break;
X
X    if (*cp == 'E' || *cp == 'F')
X    {
X         errno = atoi(cp + 1);
X         while (read(READ(fildes), &c, 1) == 1)
X              if (c == '\n')
X                   break;
```

```
X
X        if (*cp == 'F')
X             _rmt_panic(fildes);
X
X        return(-1);
X    }
X
X/*
X *  check for mis-synced pipes
X */
X
X    if (*cp != 'A')
X    {
X        _rmt_panic(fildes);
X        errno = EIO;
X        return(-1);
X    }
X
X    return(atoi(cp + 1));
X}
X
X#ifdef USE_REXEC
X
X/*
X * _rmt_rexec
X *
X * execute /etc/rmt on a remote system using rexec().
X * Return file descriptor of bidirectional socket for stdin and stdout
X * If username is NULL, or an empty string, uses current username.
X *
X * ADR: By default, this code is not used, since it requires that
X * the user have a .netrc file in his/her home directory, or that the
X * application designer be willing to have rexec prompt for login and
X * password info. This may be unacceptable, and .rhosts files for use
X * with rsh are much more common on BSD systems.
X */
X
Xstatic int
X_rmt_rexec(host, user)
Xchar *host;
Xchar *user;       /* may be NULL */
X{
X    struct servent *rexecserv;
X
X    rexecserv = getservbyname("exec", "tcp");
X    if (NULL == rexecserv) {
X        fprintf (stderr, "? exec/tcp: service not available.");
X        exit (-1);
X    }
X    if ((user != NULL) && *user == '\0')
X        user = (char *) NULL;
X    return rexec (&host, rexecserv->s_port, user, NULL,
X            "/etc/rmt", (int *)NULL);
```

```
X}
X#endif /* USE_REXEC */
X
X/*
X *  _rmt_open --- open a magtape device on system specified, as given user
X *
X *  file name has the form [user@]system:/dev/????
X#ifdef COMPAT
X *  file name has the form system[.user]:/dev/????
X#endif
X */
X
X#define MAXHOSTLEN    257  /* BSD allows very long host names... */
X
Xint __rmt_open (path, oflag, mode, bias)
Xchar *path;
Xint oflag;
Xint mode;
Xint bias;
X{
X    int i, rc;
X    char buffer[BUFMAGIC];
X    char system[MAXHOSTLEN];
X    char device[BUFMAGIC];
X    char login[BUFMAGIC];
X    char *sys, *dev, *user;
X
X    sys = system;
X    dev = device;
X    user = login;
X
X/*
X *  first, find an open pair of file descriptors
X */
X
X    for (i = 0; i < MAXUNIT; i++)
X        if (READ(i) == -1 && WRITE(i) == -1)
X            break;
X
X    if (i == MAXUNIT)
X    {
X        errno = EMFILE;
X        return(-1);
X    }
X
X/*
X *  pull apart system and device, and optional user
X *  don't munge original string
X *  if COMPAT is defined, also handle old (4.2) style person.site notation.
X */
X
X    while (*path != '@'
X#ifdef COMPAT
```

```
X             && *path != '.'
X#endif
X             && *path != ':') {
X         *sys++ = *path++;
X    }
X    *sys = '\0';
X    path++;
X
X    if (*(path - 1) == '@')
X    {
X         (void) strcpy (user, system);    /* saw user part of user@host */
X         sys = system;           /* start over */
X         while (*path != ':') {
X              *sys++ = *path++;
X         }
X         *sys = '\0';
X         path++;
X    }
X#ifdef COMPAT
X    else if (*(path - 1) == '.')
X    {
X         while (*path != ':') {
X              *user++ = *path++;
X         }
X         *user = '\0';
X         path++;
X    }
X#endif
X    else
X         *user = '\0';
X
X    while (*path) {
X         *dev++ = *path++;
X    }
X    *dev = '\0';
X
X#ifdef USE_REXEC
X/*
X *  Execute the remote command using rexec
X */
X    READ(i) = WRITE(i) = _rmt_rexec(system, login);
X    if (READ(i) < 0)
X         return -1;
X#else
X/*
X *  setup the pipes for the 'rsh' command and fork
X */
X
X    if (pipe(Ptc[i]) == -1 || pipe(Ctp[i]) == -1)
X         return(-1);
X
X    if ((rc = fork()) == -1)
X         return(-1);
```

```
X
X    if (rc == 0)
X    {
X         close(0);
X         dup(Ptc[i][0]);
X         close(Ptc[i][0]); close(Ptc[i][1]);
X         close(1);
X         dup(Ctp[i][1]);
X         close(Ctp[i][0]); close(Ctp[i][1]);
X         (void) setuid (getuid ());
X         (void) setgid (getgid ());
X         if (*user)
X         {
X              execl("/usr/ucb/rsh", "rsh", system, "-l", login,
X                   "/etc/rmt", (char *) 0);
X              execl("/usr/bin/remsh", "remsh", system, "-l", login,
X                   "/etc/rmt", (char *) 0);
X         }
X         else
X         {
X              execl("/usr/ucb/rsh", "rsh", system,
X                   "/etc/rmt", (char *) 0);
X              execl("/usr/bin/remsh", "remsh", system,
X                   "/etc/rmt", (char *) 0);
X         }
X
X/*
X *  bad problems if we get here
X */
X
X         perror("exec");
X         exit(1);
X    }
X
X    close(Ptc[i][0]); close(Ctp[i][1]);
X#endif
X
X/*
X *  now attempt to open the tape device
X */
X
X    sprintf(buffer, "O%s\n%d\n", device, oflag);
X    if (command(i, buffer) == -1 || status(i) == -1)
X         return(-1);
X
X    return(i+bias);
X}
X
X
X
X/*
X *  _rmt_close --- close a remote magtape unit and shut down
X */
```

```
X
X int __rmt_close(fildes)
Xint fildes;
X{
X    int rc;
X
X    if (command(fildes, "C\n") != -1)
X    {
X         rc = status(fildes);
X
X         _rmt_panic(fildes);
X         return(rc);
X    }
X
X    return(-1);
X}
X
X
X
X/*
X *  _rmt_read --- read a buffer from a remote tape
X */
X
Xint __rmt_read(fildes, buf, nbyte)
Xint fildes;
Xchar *buf;
Xunsigned int nbyte;
X{
X    int rc, i;
X    char buffer[BUFMAGIC];
X
X    sprintf(buffer, "R%d\n", nbyte);
X    if (command(fildes, buffer) == -1 || (rc = status(fildes)) == -1)
X         return(-1);
X
X    for (i = 0; i < rc; i += nbyte, buf += nbyte)
X    {
X         nbyte = read(READ(fildes), buf, rc);
X         if (nbyte <= 0)
X         {
X              _rmt_panic(fildes);
X              errno = EIO;
X              return(-1);
X         }
X    }
X
X    return(rc);
X}
X
X
X
X/*
X *  _rmt_write --- write a buffer to the remote tape
```

```
X */
X
Xint __rmt_write(fildes, buf, nbyte)
Xint fildes;
Xchar *buf;
Xunsigned int nbyte;
X{
X    char buffer[BUFMAGIC];
X    int (*pstat)();
X
X    sprintf(buffer, "W%d\n", nbyte);
X    if (command(fildes, buffer) == -1)
X        return(-1);
X
X    pstat = signal(SIGPIPE, SIG_IGN);
X    if (write(WRITE(fildes), buf, nbyte) == nbyte)
X    {
X        signal (SIGPIPE, pstat);
X        return(status(fildes));
X    }
X
X    signal (SIGPIPE, pstat);
X    _rmt_panic(fildes);
X    errno = EIO;
X    return(-1);
X}
X
X
X
X/*
X *  _rmt_lseek --- perform an imitation lseek operation remotely
X */
X
Xlong __rmt_lseek(fildes, offset, whence)
Xint fildes;
Xlong offset;
Xint whence;
X{
X    char buffer[BUFMAGIC];
X
X    sprintf(buffer, "L%d\n%d\n", offset, whence);
X    if (command(fildes, buffer) == -1)
X        return(-1);
X
X    return(status(fildes));
X}
X
X
X/*
X *  _rmt_ioctl --- perform raw tape operations remotely
X */
X
X#ifdef RMTIOCTL
```

```
X__rmt_ioctl(fildes, op, arg)
Xint fildes, op;
Xchar *arg;
X{
X    char c;
X    int rc, cnt;
X    char buffer[BUFMAGIC];
X
X/*
X *  MTIOCOP is the easy one. nothing is transfered in binary
X */
X
X    if (op == MTIOCTOP)
X    {
X         sprintf(buffer, "I%d\n%d\n", ((struct mtop *) arg)->mt_op,
X              ((struct mtop *) arg)->mt_count);
X         if (command(fildes, buffer) == -1)
X              return(-1);
X         return(status(fildes));
X    }
X
X/*
X *  we can only handle 2 ops, if not the other one, punt
X */
X
X    if (op != MTIOCGET)
X    {
X         errno = EINVAL;
X         return(-1);
X    }
X
X/*
X *  grab the status and read it directly into the structure
X *  this assumes that the status buffer is (hopefully) not
X *  padded and that 2 shorts fit in a long without any word
X *  alignment problems, ie - the whole struct is contiguous
X *  NOTE - this is probably NOT a good assumption.
X */
X
X    if (command(fildes, "S") == -1 || (rc = status(fildes)) == -1)
X         return(-1);
X
X    for (; rc > 0; rc -= cnt, arg += cnt)
X    {
X         cnt = read(READ(fildes), arg, rc);
X         if (cnt <= 0)
X         {
X              _rmt_panic(fildes);
X              errno = EIO;
X              return(-1);
X         }
X    }
X
```

```
X/*
X *  now we check for byte position. mt_type is a small integer field
X *  (normally) so we will check its magnitude. if it is larger than
X *  256, we will assume that the bytes are swapped and go through
X *  and reverse all the bytes
X */
X
X    if (((struct mtget *) arg)->mt_type < 256)
X         return(0);
X
X    for (cnt = 0; cnt < rc; cnt += 2)
X    {
X         c = arg[cnt];
X         arg[cnt] = arg[cnt+1];
X         arg[cnt+1] = c;
X    }
X
X    return(0);
X  }
X#endif /* RMTIOCTL */
SHAR_EOF
chmod 0644 RTAPE_LI.C || echo "restore of RTAPE_LI.C fails"
echo "x - extracting RTAPE_SE.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > RTAPE_SE.C &&
X/*
X * Copyright (c) 1983 Regents of the University of California.
X * All rights reserved.
X *
X * Redistribution and use in source and binary forms are permitted
X * provided that the above copyright notice and this paragraph are
X * duplicated in all such forms and that any documentation,
X * advertising materials, and other materials related to such
X * distribution and use acknowledge that the software was developed
X * by the University of California, Berkeley.  The name of the
X * University may not be used to endorse or promote products derived
X * from this software without specific prior written permission.
X * THIS SOFTWARE IS PROVIDED ``AS IS'' AND WITHOUT ANY EXPRESS OR
X * IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED
X * WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE.
X */
X
X#ifndef lint
Xchar copyright[] =
X"@(#) Copyright (c) 1983 Regents of the University of California.\n\
X All rights reserved.\n";
X#endif /* not lint */
X
X#ifndef lint
Xstatic char sccsid[] = "@(#)rmt.c    5.4 (Berkeley) 6/29/88";
X#endif /* not lint */
X
X/* JF added #ifdef about SO_RCVBUF in attempt to make this run on more
X   machines.  Maybe it'll work */
```

```
X/*
X * rmt
X */
X#include <stdio.h>
X#include <sgtty.h>
X#include <sys/types.h>
X#include <sys/socket.h>
X#include <sys/mtio.h>
X#include <errno.h>
X
Xint tape = -1;
X
Xchar    *record;
Xint maxrecsize = -1;
Xchar    *checkbuf();
X
X#define  SSIZE    64
Xchar    device[SSIZE];
Xchar    count[SSIZE], mode[SSIZE], pos[SSIZE], op[SSIZE];
X
Xextern  errno;
Xchar    *sys_errlist[];
Xchar    resp[BUFSIZ];
X
Xlong    lseek();
X
XFILE    *debug;
X#define  DEBUG(f)  if (debug) fprintf(debug, f)
X#define  DEBUG1(f,a)   if (debug) fprintf(debug, f, a)
X#define  DEBUG2(f,a1,a2)    if (debug) fprintf(debug, f, a1, a2)
X
Xmain(argc, argv)
X    int argc;
X    char **argv;
X{
X    int rval;
X    char c;
X    int n, i, cc;
X
X    argc--, argv++;
X    if (argc > 0) {
X         debug = fopen(*argv, "w");
X         if (debug == 0)
X              exit(1);
X         (void) setbuf(debug, (char *)0);
X    }
Xtop:
X    errno = 0;
X    rval = 0;
X    if (read(0, &c, 1) != 1)
X         exit(0);
X    switch (c) {
X
```

```
X    case 'O':
X         if (tape >= 0)
X              (void) close(tape);
X         getstring(device); getstring(mode);
X         DEBUG2("rmtd: O %s %s\n", device, mode);
X         tape = open(device, atoi(mode),0666);
X         if (tape < 0)
X              goto ioerror;
X         goto respond;
X
X    case 'C':
X         DEBUG("rmtd: C\n");
X         getstring(device);     /* discard */
X         if (close(tape) < 0)
X              goto ioerror;
X         tape = -1;
X         goto respond;
X
X    case 'L':
X         getstring(count); getstring(pos);
X         DEBUG2("rmtd: L %s %s\n", count, pos);
X         rval = lseek(tape, (long) atoi(count), atoi(pos));
X         if (rval < 0)
X              goto ioerror;
X         goto respond;
X
X    case 'W':
X         getstring(count);
X         n = atoi(count);
X         DEBUG1("rmtd: W %s\n", count);
X         record = checkbuf(record, n);
X         for (i = 0; i < n; i += cc) {
X              cc = read(0, &record[i], n - i);
X              if (cc <= 0) {
X                   DEBUG("rmtd: premature eof\n");
X                   exit(2);
X              }
X         }
X         rval = write(tape, record, n);
X         if (rval < 0)
X              goto ioerror;
X         goto respond;
X
X    case 'R':
X         getstring(count);
X         DEBUG1("rmtd: R %s\n", count);
X         n = atoi(count);
X         record = checkbuf(record, n);
X         rval = read(tape, record, n);
X         if (rval < 0)
X              goto ioerror;
X         (void) sprintf(resp, "A%d\n", rval);
X         (void) write(1, resp, strlen(resp));
```

```
X         (void) write(1, record, rval);
X         goto top;
X
X    case 'I':
X         getstring(op); getstring(count);
X         DEBUG2("rmtd: I %s %s\n", op, count);
X         { struct mtop mtop;
X           mtop.mt_op = atoi(op);
X           mtop.mt_count = atoi(count);
X           if (ioctl(tape, MTIOCTOP, (char *)&mtop) < 0)
X              goto ioerror;
X           rval = mtop.mt_count;
X         }
X         goto respond;
X
X    case 'S':     /* status */
X         DEBUG("rmtd: S\n");
X         { struct mtget mtget;
X           if (ioctl(tape, MTIOCGET, (char *)&mtget) < 0)
X              goto ioerror;
X           rval = sizeof (mtget);
X           (void) sprintf(resp, "A%d\n", rval);
X           (void) write(1, resp, strlen(resp));
X           (void) write(1, (char *)&mtget, sizeof (mtget));
X           goto top;
X         }
X
X    default:
X         DEBUG1("rmtd: garbage command %c\n", c);
X         exit(3);
X    }
Xrespond:
X    DEBUG1("rmtd: A %d\n", rval);
X    (void) sprintf(resp, "A%d\n", rval);
X    (void) write(1, resp, strlen(resp));
X    goto top;
Xioerror:
X    error(errno);
X    goto top;
X}
X
Xgetstring(bp)
X    char *bp;
X{
X    int i;
X    char *cp = bp;
X
X    for (i = 0; i < SSIZE; i++) {
X         if (read(0, cp+i, 1) != 1)
X              exit(0);
X         if (cp[i] == '\n')
X              break;
X    }
```

```
X    cp[i] = '\0';
X}
X
Xchar *
Xcheckbuf(record, size)
X    char *record;
X    int size;
X{
X    extern char *malloc();
X
X    if (size <= maxrecsize)
X         return (record);
X    if (record != 0)
X         free(record);
X    record = malloc(size);
X    if (record == 0) {
X         DEBUG("rmtd: cannot allocate buffer space\n");
X         exit(4);
X    }
X    maxrecsize = size;
X#ifdef SO_RCVBUF
X    while (size > 1024 &&
X           setsockopt(0, SOL_SOCKET, SO_RCVBUF, &size, sizeof (size)) < 0)
X         size -= 1024;
X#else
X    size= 1+((size-1)%1024);
X#endif
X    return (record);
X}
X
Xerror(num)
X    int num;
X{
X
X    DEBUG2("rmtd: E %d (%s)\n", num, sys_errlist[num]);
X    (void) sprintf(resp, "E%d\n%s\n", num, sys_errlist[num]);
X    (void) write(1, resp, strlen (resp));
X}
SHAR_EOF
chmod 0644 RTAPE_SE.C || echo "restore of RTAPE_SE.C fails"
echo "x - extracting TAR.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > TAR.C &&
X/*
X
X    Copyright (C) 1988 Free Software Foundation
X
XGNU tar is distributed in the hope that it will be useful, but WITHOUT ANY
XWARRANTY.  No author or distributor accepts responsibility to anyone
Xfor the consequences of using it or for whether it serves any
Xparticular purpose or works at all, unless he says so in writing.
XRefer to the GNU tar General Public License for full details.
X
XEveryone is granted permission to copy, modify and redistribute GNU tar,
```

```
Xbut only under the conditions described in the GNU tar General Public
XLicense.  A copy of this license is supposed to have been given to you
Xalong with GNU tar so you can know your rights and responsibilities.  It
Xshould be in a file named COPYING.  Among other things, the copyright
Xnotice and this notice must be preserved on all copies.
X
XIn other words, go ahead and share GNU tar, but don't try to stop
Xanyone else from sharing it farther.  Help stamp out software hoarding!
X*/
X
X/*
X * A tar(1) program.
X *
X * Written by John Gilmore, ihnp4!hoptoad!gnu, starting 25 Aug 85.
X *
X * @(#)tar.c 1.34 11/6/87 - gnu
X */
X
X#include <stdio.h>
X#ifdef atarist
X#include <types.h>         /* Needed for typedefs in tar.h */
X#include <stat.h>     /* JF */
X#else
X#include <sys/types.h>     /* Needed for typedefs in tar.h */
X#include <sys/stat.h>      /* JF */
X#endif
X
X#ifdef USG
X#define rindex strrchr
X#endif
X
X#ifdef BSD42
X#include <sys/dir.h>
X#else
X#if (defined(MSDOS) && !defined(atarist))
X#include <sys/dir.h>
X#else
X#ifdef USG
X#include <dirent.h>
X#define direct dirent
X#define DP_NAMELEN(x) strlen((x)->d_name)
X#else
X/*
X * FIXME: On other systems there is no standard place for the header file
X * for the portable directory access routines.  Change the #include line
X * below to bring it in from wherever it is.
X */
X#include <dirent.h>
X#define direct dirent      /* if you have POSIX routines */
X#define DP_NAMELEN(x) strlen((x)->d_name)  /* ditto */
X#endif
X#endif
X#endif
```

```
X
X#ifndef DP_NAMELEN
X#define DP_NAMELEN(x)  (x)->d_namlen
X#endif
X
Xextern char  *malloc();
Xextern char  *getenv();
Xextern char  *strncpy();
Xextern char  *index();
Xextern char  *strcpy();    /* JF */
Xextern char  *strcat();    /* JF */
X
Xextern char  *optarg; /* Pointer to argument */
Xextern int   optind;       /* Global argv index from getopt */
X
X/*
X * The following causes "tar.h" to produce definitions of all the
X * global variables, rather than just "extern" declarations of them.
X */
X#define TAR_EXTERN /**/
X#include "tar.h"
X
X/*
X * We should use a conversion routine that does reasonable error
X * checking -- atoi doesn't.  For now, punt.  FIXME.
X */
X#define intconv   atoi
Xextern int   getoldopt();
Xextern void  read_and();
Xextern void  list_archive();
Xextern void  extract_archive();
Xextern void  diff_archive();
Xextern void  create_archive();
Xextern void  update_archive();
Xextern void  junk_archive();
X
X/* JF */
Xextern time_t    getdate();
X#ifdef __GNU__
Xextern void *init_buffer();
X#else
Xextern char *init_buffer();
X#endif
Xextern char *get_buffer();
Xextern void add_buffer();
Xextern void flush_buffer();
X
Xtime_t new_time;
X
Xstatic FILE  *namef;       /* File to read names from */
Xstatic char  **n_argv; /* Argv used by name routines */
Xstatic int   n_argc;  /* Argc used by name routines */
X                 /* They also use "optind" from getopt(). */
```

```
X
Xvoid    describe();
Xvoid    options();
X
X#ifndef S_IFLNK
X#define lstat stat
X#endif
X
X#ifndef DEFBLOCKING
X#define DEFBLOCKING 20
X#endif
X
X#ifndef DEF_AR_FILE
X#define DEF_AR_FILE "tar.out"
X#endif
X
X#ifdef atarist
Xint setmode() { return 0; } /* MSDOS compatibility cruft */
X#endif
X
X/*
X * Main routine for tar.
X */
Xmain(argc, argv)
X    int  argc;
X    char **argv;
X{
X#ifdef atarist
X    extern char *name_next();
X    extern int _unixmode;
X    static char archname[NAMSIZ+2];
X#endif
X    /* Uncomment this message in particularly buggy versions...
X    fprintf(stderr,
X     "tar: You are running an experimental PD tar, maybe use /bin/tar.\n");
X     */
X
X#ifdef atarist
X    _unixmode = 3;          /* convert filenames to/from Un*x */
X    tar = *argv[0] ? argv[0] : "tar";
X#else
X    tar = argv[0];          /* JF: was "tar" Set program name */
X#endif
X    options(argc, argv);
X
X    name_init(argc, argv);
X#ifdef atarist
X    if (ar_file == 0) {
X         strcpy(archname, name_next(0));
X         ar_file = archname;
X    }
X#endif
X    switch(cmd_mode) {
```

```
X    case CMD_CAT:
X    case CMD_UPDATE:
X    case CMD_APPEND:
X         update_archive();
X         break;
X    case CMD_DELETE:
X         junk_archive();
X         break;
X    case CMD_CREATE:
X         create_archive();
X         break;
X    case CMD_EXTRACT:
X         extr_init();
X         read_and(extract_archive);
X         break;
X    case CMD_LIST:
X         read_and(list_archive);
X         break;
X    case CMD_DIFF:
X         diff_init();
X         read_and(diff_archive);
X         break;
X    case CMD_NONE:
X         fprintf (stderr,
X#ifdef atarist
X"%s: you must specify exactly one of the r, c, t, x, or d options\n", tar);
X#else
X"tar: you must specify exactly one of the r, c, t, x, or d options\n");
X#endif
X         describe();
X         exit(EX_ARGSBAD);
X    }
X    exit(0);
X    /* NOTREACHED */
X}
X
X
X/*
X * Parse the options for tar.
X */
Xvoid
Xoptions(argc, argv)
X    int  argc;
X    char **argv;
X{
X    register int  c;       /* Option letter */
X
X    /* Set default option values */
X    blocking = DEFBLOCKING;     /* From Makefile */
X    ar_file = getenv("TAPE");   /* From environment, or */
X    if (ar_file == 0)
X         ar_file = DEF_AR_FILE; /* From Makefile */
X
```

```
X    /* Parse options */
X    while ((c = getoldopt(argc, argv,
"01234567Ab:BcdDf:GhikK:lmMN:oOprRstT:uvV:wWxX:zZ")
X         ) != EOF) {
X         switch (c) {
X         case '0':
X         case '1':
X         case '2':
X         case '3':
X         case '4':
X         case '5':
X         case '6':
X         case '7':
X              {
X                   /* JF this'll have to be modified for other
X                      systems, of course! */
X#ifndef LOW_NUM
X#define LOW_NUM 0
X#define MID_NUM 8
X#define HGH_NUM 16
X#endif
X                   int d,add;
X                   static char buf[50];
X
X                   d=getoldopt(argc,argv,"lmh");
X                   if(d=='l') add=LOW_NUM;
X                   else if(d=='m') add=MID_NUM;
X                   else if(d=='h') add=HGH_NUM;
X                   else goto badopt;
X                   sprintf(buf,"/dev/rmt%d",add+c-'0');
X                   ar_file=buf;
X              }
X              break;
X
X         case 'A':          /* Arguments are tar files,
X                               just cat them onto the end
X                               of the archive.  */
X              if(cmd_mode!=CMD_NONE)
X                   goto badopt;
X              cmd_mode=CMD_CAT;
X              break;
X
X         case 'b':          /* Set blocking factor */
X              blocking = intconv(optarg);
X              break;
X
X         case 'B':          /* Try to reblock input */
X              f_reblock++;       /* For reading 4.2BSD pipes */
X              break;
X
X         case 'c':          /* Create an archive */
X              if(cmd_mode!=CMD_NONE)
X                   goto badopt;
```

```
X             cmd_mode=CMD_CREATE;
X             break;
X
X        case 'd':          /* Find difference tape/disk */
X             if(cmd_mode!=CMD_NONE)
X                  goto badopt;
X             cmd_mode=CMD_DIFF;
X             break;
X
X        case 'D':          /* Delete in the archive */
X             if(cmd_mode!=CMD_NONE)
X                  goto badopt;
X             cmd_mode=CMD_DELETE;
X             break;
X
X        case 'f':          /* Use ar_file for the archive */
X             ar_file = optarg;
X             break;
X
X        case 'G':          /* We are making a GNU dump; save
X                              directories at the beginning of
X                              the archive, and include in each
X                              directory its contents */
X             if(f_oldarch)
X                  goto badopt;
X             f_gnudump++;
X             break;
X
X        case 'h':
X             if(optind && argv[optind] && !strcmp(argv[optind],"-help")) {
X                  printf("This is GNU tar, the tape archiving program.\n");
X                  describe();
X                  printf("For a more complete description of how to use tar, type
'info util tar'\n");
X                  exit(1);
X             } else
X                  f_follow_links++;  /* follow symbolic links */
X             break;
X
X        case 'i':
X             f_ignorez++;       /* Ignore zero records (eofs) */
X             /*
X              * This can't be the default, because Unix tar
X              * writes two records of zeros, then pads out the
X              * block with garbage.
X              */
X             break;
X
X        case 'k':          /* Don't overwrite files */
X#ifdef NO_OPEN3
X             fprintf(stderr,
X                  "tar: can't do -k option on this system\n");
X             exit(EX_ARGSBAD);
```

```
X#else
X             f_keep++;
X#endif
X             break;
X
X        case 'K':
X             f_startfile++;
X             addname(optarg);
X             break;
X
X        case 'l':          /* When dumping directories, don't
X                              dump files/subdirectories that are
X                              on other filesystems. */
X             f_local_filesys++;
X             break;
X
X        case 'm':
X             f_modified++;
X             break;
X
X        case 'M':          /* Make Multivolume archive:
X                              When we can't write any more
X                              into the archive, re-open it,
X                              and continue writing */
X             f_multivol++;
X             break;
X
X        case 'N':          /* Only write files newer than X */
X             f_new_files++;
X             new_time=getdate(optarg,(struct timeb *)0);
X             break;
X
X        case 'o':          /* Generate old archive */
X             if(f_gnudump /* || f_dironly */)
X                  goto badopt;
X             f_oldarch++;
X             break;
X
X        case 'O':
X             f_exstdout++;
X             break;
X
X        case 'p':
X             f_use_protection++;
X             break;
X
X        case 'r':          /* Append files to the archive */
X             if(cmd_mode!=CMD_NONE)
X                  goto badopt;
X             cmd_mode=CMD_APPEND;
X             break;
X
X        case 'R':
```

```
X             f_sayblock++;      /* Print block #s for debug */
X             break;             /* of bad tar archives */
X
X        case 's':
X             f_sorted_names++;  /* Names to extr are sorted */
X             break;
X
X        case 't':
X             if(cmd_mode!=CMD_NONE)
X                  goto badopt;
X             cmd_mode=CMD_LIST;
X             f_verbose++;       /* "t" output == "cv" or "xv" */
X             break;
X
X        case 'T':
X             name_file = optarg;
X             f_namefile++;
X             break;
X
X        case 'u':          /* Append files to the archive that
X                              aren't there, or are newer than the
X                              copy in the archive */
X             if(cmd_mode!=CMD_NONE)
X                  goto badopt;
X             cmd_mode=CMD_UPDATE;
X             break;
X
X        case 'v':
X             if(optind && argv[optind] && !strcmp(argv[optind],"-version")) {
X                  extern char version_string[];
X
X                  fprintf(stderr,"%s\n",version_string);
X                  while(getoldopt(argc,argv,"ersion")!='n')
X                       ;
X             } else {
X                  f_verbose++;
X             }
X             break;
X
X        case 'V':
X             f_volhdr=optarg;
X             break;
X
X        case 'w':
X             f_confirm++;
X             break;
X
X        case 'W':
X             f_verify++;
X             break;
X
X        case 'x':          /* Extract files from the archive */
X             if(cmd_mode!=CMD_NONE)
```

```
X                  goto badopt;
X             cmd_mode=CMD_EXTRACT;
X             break;
X
X         case 'X':
X             f_exclude++;
SHAR_EOF
echo "End of part 5"
echo "File TAR.C is continued in part 6"
echo "6" > s2_seq_.tmp
exit 0


From -3232422943035372046
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,fa19da2d52766c94,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-19 14:43:33 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
decwrl!sun-barr!newstop!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i052: gnutar -- GNU's version of UNIX "tar"  part06/08
Keywords: shar, C
Message-ID: <33892@grapevine.uucp>
Date: 19 Jul 89 21:43:33 GMT
Sender: news@grapevine.uucp
Lines: 1564
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: 7103_300@uwovax.uwo.ca (Eric R. Smith)
Posting-number: Volume 2, Issue 52
Archive-name: gnutar/part06

#!/bin/sh
# this is part 6 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file TAR.C continued
#
CurArch=6
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file TAR.C"
sed 's/^X//' << 'SHAR_EOF' >> TAR.C
X             add_exclude(optarg);
X             break;
```

```
X
X        case 'z':     /* Easy to type */
X        case 'Z':     /* Like the filename extension .Z */
X             f_compress++;
X             break;
X
X        case '?':
X        badopt:
X             describe();
X             exit(EX_ARGSBAD);
X
X        }
X    }
X
X    blocksize = blocking * RECORDSIZE;
X}
X
X
X/*
X * Print as much help as the user's gonna get.
X *
X * We have to sprinkle in the KLUDGE lines because too many compilers
X * cannot handle character strings longer than about 512 bytes.  Yuk!
X * In particular, MSDOS MSC 4.0 (and 5.0) and PDP-11 V7 Unix have this
X * problem.
X */
Xvoid
Xdescribe()
X{
X    fprintf(stderr,"%s: choose one of the following:\n",tar);
X    fputs("\
X-A  Append tar files to an archive.\n\
X-c  Create a new archive.\n\
X-d  Find differences between archive and file system.\n\
X-D  Delete from the archive (NOT for use on mag tapes!)\n\
X-r  Append files to the end of an archive.\n\
X-t  List the contents of an archive.\n\
X-u  Only append files that are newer than the copy in an archive.\n\
X-x  Extract files from an archive.\n",stderr);
X
X    fputs("\
XOther options:\n\
X-b N    blocking factor N (block size = Nx512 bytes)\n\
X-B  reblock as we read (for reading 4.2BSD pipes)\n\
X-C dir  change to directory DIR\n\
X", stderr); /* KLUDGE */ fputs("\
X-f F    read/write archive from file or device F (or hostname:/ForD)\n\
X-G  create/list/extract GNU-format backup\n\
X-h  don't dump symbolic links; dump the files they point to\n\
X-i  ignore blocks of zeros in the archive, which normally mean EOF\n\
X-k  keep existing files, don't overwrite them from the archive\n\
X-K file  begin at \"file\" in the archive\n\
X-l  stay in the local file system (like dump(8)) when creating an archive\n\
```

```
X", stderr); /* KLUDGE */ fputs("\
X-m  don't extract file modified time\n\
X-M  create/list/extract multi-volume archive\n\
X-N date  only store files newer than DATE\n\
X-o  write an old V7 format archive, rather than ANSI [draft 6] format\n\
X-O  extract files to standard output\n\
X-p  do extract all protection information\n\
X-R  dump record number within archive with each message\n\
X-s  list of names to extract is sorted to match the archive\n\
X", stderr); /* KLUDGE */ fputs("\
X-T F    get names to extract or create from file F\n\
X-v  verbosely list what files we process\n\
X-V vnam  create archive with volume name 'vnam'\n\
X-w  ask for confirmation for every action\n\
X-W  attempt to verify the archive after writing it\n\
X-X file  eXclude files listed in \"file\"\n\
X-z or Z  run the archive through compress(1)\n\
X-[0-7][lmh] specify drive and density\n\
X", stderr);
X}
X
X
X/*
X * Set up to gather file names for tar.
X *
X * They can either come from stdin or from argv.
X */
Xname_init(argc, argv)
X    int  argc;
X    char **argv;
X{
X
X    if (f_namefile) {
X         if (optind < argc) {
X              fprintf(stderr, "tar: too many args with -T option\n");
X              exit(EX_ARGSBAD);
X         }
X         if (!strcmp(name_file, "-")) {
X              namef = stdin;
X         } else {
X              namef = fopen(name_file, "r");
X              if (namef == NULL) {
X                   msg_perror("can't open file %s",name_file);
X                   exit(EX_BADFILE);
X              }
X         }
X    } else {
X         /* Get file names from argv, after options. */
X         n_argc = argc;
X         n_argv = argv;
X    }
X}
X
```

```
X/*
X * Get the next name from argv or the name file.
X *
X * Result is in static storage and can't be relied upon across two calls.
X */
X
X/* C is non-zero if we should deal with -C */
Xchar *
Xname_next(c)
X{
X    static char   buffer[NAMSIZ+2];  /* Holding pattern */
X    register char *p;
X    register char *q;
X    extern char *un_quote_string();
X
X tryagain:
X    if (namef == NULL) {
X         /* Names come from argv, after options */
X         if (optind < n_argc) {
X              /* JF trivial support for -C option.  I don't know if
X                 chdir'ing at this point is dangerous or not.
X                 It seems to work, which is all I ask. */
X              if(c && n_argv[optind][0]=='-' && n_argv[optind][1]=='C' &&
n_argv[optind][2]=='\0') {
X                   optind++;
X                   if(chdir(n_argv[optind])<0) {
X                        msg_perror("Can't chdir to %s",n_argv[optind]);
X                   }
X                   optind++;
X              }
X              /* End of JF quick -C hack */
X
X              if(f_exclude && check_exclude(n_argv[optind])) {
X                   optind++;
X                   goto tryagain;
X              }
X              return un_quote_string(n_argv[optind++]);
X         }
X         return (char *)NULL;
X    }
X    while(p = fgets(buffer, NAMSIZ+1 /*nl*/, namef)) {
X         q = p+strlen(p)-1;      /* Find the newline */
X         if (q <= p)             /* Ignore empty lines */
X              continue;
X         *q-- = '\0';            /* Zap the newline */
X         while (q > p && *q == '/')  /* Zap trailing /s */
X              *q-- = '\0';
X         if(f_exclude && check_exclude(p))
X              goto tryagain;
X         return un_quote_string(p);
X    }
X    return NULL;
X}
```

```
X
X
X/*
X * Close the name file, if any.
X */
Xname_close()
X{
X
X    if (namef != NULL && namef != stdin) fclose(namef);
X}
X
X
X/*
X * Gather names in a list for scanning.
X * Could hash them later if we really care.
X *
X * If the names are already sorted to match the archive, we just
X * read them one by one.  name_gather reads the first one, and it
X * is called by name_match as appropriate to read the next ones.
X * At EOF, the last name read is just left in the buffer.
X * This option lets users of small machines extract an arbitrary
X * number of files by doing "tar t" and editing down the list of files.
X */
Xname_gather()
X{
X    register char *p;
X    static struct name namebuf[1];   /* One-name buffer */
X    static char *chdir_name;
X
X    if (f_sorted_names) {
X         p = name_next(0);
X         if (p) {
X              if(*p=='-' && p[1]=='C' && p[2]=='\0') {
X                   chdir_name=name_next(0);
X                   p=name_next(0);
X                   if(!p) {
X                        fprintf(stderr,"Missing file name after -C\n");
X                        exit(EX_ARGSBAD);
X                   }
X                   namebuf[0].change_dir=chdir_name;
X              }
X              namebuf[0].length = strlen(p);
X              if (namebuf[0].length >= sizeof namebuf[0].name) {
X                   fprintf(stderr, "Argument name too long: %s\n",
X                        p);
X                   namebuf[0].length = (sizeof namebuf[0].name) - 1;
X              }
X              strncpy(namebuf[0].name, p, namebuf[0].length);
X              namebuf[0].name[ namebuf[0].length ] = 0;
X              namebuf[0].next = (struct name *)NULL;
X              namebuf[0].found = 0;
X              namelist = namebuf;
X              namelast = namelist;
```

```
X         }
X         return;
X    }
X
X    /* Non sorted names -- read them all in */
X    while (p = name_next(0))
X         addname(p);
X}
X
X/*
X * Add a name to the namelist.
X */
Xaddname(name)
X    char *name;             /* pointer to name */
X{
X    register int  i;        /* Length of string */
X    register struct name   *p;  /* Current struct pointer */
X    static char *chdir_name;
X    char *new_name();
X#define MAXPATHLEN 1024
X
X    if(name[0]=='-' && name[1]=='C' && name[2]=='\0') {
X         chdir_name=name_next(0);
X         name=name_next(0);
X         if(!name) {
X              fprintf(stderr,"Missing file name after -C\n");
X              exit(EX_ARGSBAD);
X         }
X         if(chdir_name[0]!='/') {
X              char path[MAXPATHLEN];
X#ifdef MSDOS
X              int getcwd();
X
X              if(!getcwd(path,MAXPATHLEN))
X                   fprintf(stderr,"Couldn't get current dir\n");
X                   exit(EX_SYSTEM);
X#else
X              char *getwd();
X
X              if(!getwd(path)) {
X                   fprintf(stderr,"Couldn't get current dir: %s\n",path);
X                   exit(EX_SYSTEM);
X              }
X#endif
X              chdir_name=new_name(path,chdir_name);
X         }
X    }
X
X    i = strlen(name);
X    /*NOSTRICT*/
X    p = (struct name *)
X         malloc((unsigned)(i + sizeof(struct name) - NAMSIZ));
X    if (!p) {
```

```
X         fprintf(stderr,"tar: cannot allocate mem for name %s\n",name);
X         exit(EX_SYSTEM);
X    }
X    p->next = (struct name *)NULL;
X    p->length = i;
X    strncpy(p->name, name, i);
X    p->name[i] = '\0'; /* Null term */
X    p->found = 0;
X    p->regexp = 0;          /* Assume not a regular expression */
X    p->firstch = 1;         /* Assume first char is literal */
X    p->change_dir=chdir_name;
X    p->dir_contents = 0;    /* JF */
X    if (index(name, '*') || index(name, '[') || index(name, '?')) {
X         p->regexp = 1;     /* No, it's a regexp */
X         if (name[0] == '*' || name[0] == '[' || name[0] == '?')
X              p->firstch = 0;         /* Not even 1st char literal */
X    }
X
X    if (namelast) namelast->next = p;
X    namelast = p;
X    if (!namelist) namelist = p;
X}
X
Xadd_dir_to_name(name,dirc)
Xchar *name;
Xchar *dirc;
X{
X    struct name *n;
X
X    for(n=namelist;n;n=n->next) {
X         if(!strcmp(n->name,name)) {
X              n->dir_contents = dirc;
X              return;
X         }
X    }
X}
X
X/*
X * Match a name from an archive, p, with a name from the namelist.
X */
Xname_match(p)
X    register char *p;
X{
X    register struct name   *nlp;
X    register int      len;
X
Xagain:
X    if (0 == (nlp = namelist))  /* Empty namelist is easy */
X         return 1;
X    len = strlen(p);
X    for (; nlp != 0; nlp = nlp->next) {
X         /* If first chars don't match, quick skip */
X         if (nlp->firstch && nlp->name[0] != p[0])
```

```
X             continue;
X
X        /* Regular expressions */
X        if (nlp->regexp) {
X             if (wildmat(p, nlp->name)) {
X                  nlp->found = 1;    /* Remember it matched */
X                  if(f_startfile) {
X                       free((void *)namelist);
X                       namelist=0;
X                  }
X                  return 1; /* We got a match */
X             }
X             continue;
X        }
X
X        /* Plain Old Strings */
X        if (nlp->length <= len      /* Archive len >= specified */
X         && (p[nlp->length] == '\0' || p[nlp->length] == '/')
X                         /* Full match on file/dirname */
X         && strncmp(p, nlp->name, nlp->length) == 0) /* Name compare */
X        {
X             nlp->found = 1;        /* Remember it matched */
X             if(f_startfile) {
X                  free((void *)namelist);
X                  namelist = 0;
X             }
X             return 1;     /* We got a match */
X        }
X    }
X
X    /*
X     * Filename from archive not found in namelist.
X     * If we have the whole namelist here, just return 0.
X     * Otherwise, read the next name in and compare it.
X     * If this was the last name, namelist->found will remain on.
X     * If not, we loop to compare the newly read name.
X     */
X    if (f_sorted_names && namelist->found) {
X         name_gather();         /* Read one more */
X         if (!namelist->found) goto again;
X    }
X    return 0;
X}
X
X
X/*
X * Print the names of things in the namelist that were not matched.
X */
Xnames_notfound()
X{
X    register struct name   *nlp;
X    register char     *p;
X
```

```
X    for (nlp = namelist; nlp != 0; nlp = nlp->next) {
X         if (!nlp->found) {
X              fprintf(stderr, "tar: %s not found in archive\n",
X                   nlp->name);
X         }
X         /*
X          * We could free() the list, but the process is about
X          * to die anyway, so save some CPU time.  Amigas and
X          * other similarly broken software will need to waste
X          * the time, though.
X          */
X#ifndef unix
X         if (!f_sorted_names)
X              free(nlp);
X#endif
X    }
X    namelist = (struct name *)NULL;
X    namelast = (struct name *)NULL;
X
X    if (f_sorted_names) {
X         while (0 != (p = name_next(1)))
X              fprintf(stderr, "tar: %s not found in archive\n", p);
X    }
X}
X
X/* These next routines were created by JF */
X
Xname_expand()
X{
X;
X}
X/* p is a directory.  Add all the files in P to the namelist.  If any of the
X   files is a directory, recurse on the subdirectory. . . */
Xstatic
Xadd_dir_name(p,device)
Xchar *p;
Xint device;
X{
X    char *new_buf;
X    char *p_buf;
X    char *get_dir_contents();
X    new_buf=get_dir_contents(p,device);
X
X    add_dir_to_name(p,new_buf);
X    {
X         char namebuf[NAMSIZ+2];
X         register int len;
X
X         (void)strcpy(namebuf,p);
X         len=strlen(namebuf);
X         if(namebuf[len-1]!='/') {
X              namebuf[len++]='/';
X              namebuf[len]='\0';
```

```
X        }
X        for(p_buf=new_buf;*p_buf;p_buf+=strlen(p_buf)+1) {
X             if(*p_buf=='D') {
X                  (void)strcpy(namebuf+len,p_buf+1);
X                  addname(namebuf);
X                  add_dir_name(namebuf,device);
X             }
X        }
X    }
X}
X
Xchar *
Xget_dir_contents(p,device)
Xchar *p;
Xint device;
X{
X    register DIR *dirp;
X    register struct direct *d;
X    char namebuf[NAMSIZ+2];
X    register int len;
X    extern int errno;
X
X#ifdef  __GNU__
X    void *the_buffer;
X#else
X    char *the_buffer;
X#endif
X        char *buf,*p_buf;
X        char **vec,**p_vec;
X        int n_strs,n_size;
X        char *new_buf;
X        int dirent_cmp();
X
X    if(f_local_filesys && device<0) {
X        struct stat hs;
X
X        if (0 != f_follow_links ? stat(p, &hs) : lstat(p, &hs))
X             msg_perror("can't stat %s",namebuf);
X        else
X             device=hs.st_dev;
X    }
X    errno=0;
X    dirp=opendir(p);
X    if(!dirp) {
X        if(errno)
X             msg_perror("can't open directory %s",p);
X        else
X             msg("error opening directory %s",p);
X        return "\0\0\0\0";
X    }
X    (void) strcpy(namebuf,p);
X    if(p[strlen(p)-1]!='/')
X        (void) strcat(namebuf,"/");
```

```
X    len=strlen(namebuf);
X
X    the_buffer=init_buffer();
X    while(d=readdir(dirp)) {
X         struct stat hs;
X
X         /* Skip . and .. */
X         if(is_dot_or_dotdot(d->d_name))
X              continue;
X         if(DP_NAMELEN(d) + len >=NAMSIZ) {
X              msg("%s%s name too long: skipped",namebuf,d->d_name);
X              continue;
X         }
X         (void) strcpy(namebuf+len,d->d_name);
X         if (0 != f_follow_links? stat(namebuf, &hs): lstat(namebuf, &hs)) {
X              msg_perror("can't stat %s",namebuf);
X              continue;
X         }
X         if((f_new_files && new_time>hs.st_mtime &&
X             new_time>hs.st_ctime && (hs.st_mode&S_IFMT)!=S_IFDIR)
X            || (f_local_filesys && device>=0 && device!=hs.st_dev))
X              add_buffer(the_buffer,"N",1);
X         else if((hs.st_mode&S_IFMT)==S_IFDIR)
X              add_buffer(the_buffer,"D",1);
X         else
X              add_buffer(the_buffer,"Y",1);
X         add_buffer(the_buffer,d->d_name,(int)(DP_NAMELEN(d)+1));
X    }
X    add_buffer(the_buffer,"\000\000",2);
X    closedir(dirp);
X
X    /* Well, we've read in the contents of the dir, now sort them */
X    buf=get_buffer(the_buffer);
X    n_strs=0;
X    n_size=0;
X    for(p_buf=buf;*p_buf;) {
X         int tmp;
X
X         tmp=strlen(p_buf)+1;
X         n_strs++;
X         n_size+=tmp;
X         p_buf+=tmp;
X    }
X    if(n_strs==0) {
X         flush_buffer(the_buffer);
X         return "\0\0\0\0";
X    }
X    vec=(char **)malloc(sizeof(char *)*(n_strs+1));
X    for(p_vec=vec,p_buf=buf;*p_buf;p_buf+=strlen(p_buf)+1)
X         *p_vec++= p_buf;
X    *p_vec= 0;
X    qsort((void *)vec,n_strs,sizeof(char *),dirent_cmp);
X    new_buf=(char *)malloc(n_size+2);
```

```
X    for(p_vec=vec,p_buf=new_buf;*p_vec;p_vec++) {
X         char *p_tmp;
X
X         for(p_tmp= *p_vec;*p_buf++= *p_tmp++;)
X              ;
X    }
X    *p_buf++='\0';
X    free(vec);
X    flush_buffer(the_buffer);
X    return new_buf;
X}
X
Xint dirent_cmp(p1,p2)
Xchar **p1,**p2;
X{
X    char *frst,*scnd;
X
X    frst= (*p1)+1;
X    scnd= (*p2)+1;
X
X    return strcmp(frst,scnd);
X}
X
X/* This is like name_match(), except that it returns a pointer to the name
X   it matched, and doesn't set ->found  The caller will have to do that
X   if it wants to.  Oh, and if the namelist is empty, it returns 0, unlike
X   name_match(), which returns TRUE */
X
Xstruct name *
Xname_scan(p)
Xregister char *p;
X{
X    register struct name   *nlp;
X    register int      len;
X
Xagain:
X    if (0 == (nlp = namelist))  /* Empty namelist is easy */
X         return 0;
X    len = strlen(p);
X    for (; nlp != 0; nlp = nlp->next) {
X         /* If first chars don't match, quick skip */
X         if (nlp->firstch && nlp->name[0] != p[0])
X              continue;
X
X         /* Regular expressions */
X         if (nlp->regexp) {
X              if (wildmat(p, nlp->name))
X                   return nlp;   /* We got a match */
X              continue;
X         }
X
X         /* Plain Old Strings */
X         if (nlp->length <= len      /* Archive len >= specified */
```

```
X         && (p[nlp->length] == '\0' || p[nlp->length] == '/')
X                         /* Full match on file/dirname */
X         && strncmp(p, nlp->name, nlp->length) == 0) /* Name compare */
X             return nlp;        /* We got a match */
X    }
X
X    /*
X     * Filename from archive not found in namelist.
X     * If we have the whole namelist here, just return 0.
X     * Otherwise, read the next name in and compare it.
X     * If this was the last name, namelist->found will remain on.
X     * If not, we loop to compare the newly read name.
X     */
X    if (f_sorted_names && namelist->found) {
X         name_gather();         /* Read one more */
X         if (!namelist->found) goto again;
X    }
X    return (struct name *) 0;
X}
X
X/* This returns a name from the namelist which doesn't have ->found set.
X   It sets ->found before returning, so successive calls will find and return
X   all the non-found names in the namelist */
X
Xstruct name *gnu_list_name;
X
Xchar *
Xname_from_list()
X{
X    if(!gnu_list_name)
X         gnu_list_name = namelist;
X    while(gnu_list_name && gnu_list_name->found)
X         gnu_list_name=gnu_list_name->next;
X    if(gnu_list_name) {
X         gnu_list_name->found++;
X         if(gnu_list_name->change_dir)
X              if(chdir(gnu_list_name->change_dir)<0)
X                   msg_perror("can't chdir to %s",gnu_list_name->change_dir);
X         return gnu_list_name->name;
X    }
X    return (char *)0;
X}
X
Xblank_name_list()
X{
X    struct name *n;
X
X    gnu_list_name = 0;
X    for(n=namelist;n;n=n->next)
X         n->found = 0;
X}
X
X/* Collect all the names from argv[] (or whatever), then expand them into
```

```
X   a directory tree, and put all the directories at the beginning. */
Xcollect_and_sort_names()
X{
X    struct name *n,*n_next;
X    int num_names;
X    int name_cmp();
X    char *merge_sort();
X
X    name_gather();
X
X    if(!namelist) addname(".");
X    for(n=namelist;n;n=n_next) {
X         n_next=n->next;
X         if(n->found || n->dir_contents)
X              continue;
X         if(n->regexp)      /* FIXME just skip regexps for now */
X              continue;
X         if(n->change_dir)
X              if(chdir(n->change_dir)<0) {
X                   msg_perror("can't chdir to %s",n->change_dir);
X                   continue;
X              }
X
X         if(is_a_directory(n->name)) {
X              n->found++;
X              add_dir_name(n->name,-1);
X         }
X    }
X
X    num_names=0;
X    for(n=namelist;n;n=n->next)
X         num_names++;
X    namelist=(struct name *)merge_sort((void *)namelist,num_names,(char *)
(&(namelist->next))-(char *)namelist,name_cmp);
X
X    for(n=namelist;n;n=n->next) {
X         n->found=0;
X    }
X}
X
Xint name_cmp(n1,n2)
Xstruct name *n1,*n2;
X{
X    if(n1->found) {
X         if(n2->found)
X              return strcmp(n1->name,n2->name);
X         else
X              return -1;
X    } else if(n2->found)
X         return 1;
X    else
X         return strcmp(n1->name,n2->name);
X}
```

```
X
Xchar *new_name();
X
Xgnu_restore(name)
Xchar *name;
X{
X    char *current_dir;
X/*  int current_dir_length; */
X
X    char *archive_dir;
X/*  int archive_dir_length; */
X#ifdef __GNU__
X    void *the_buffer;
X#else
X    char *the_buffer;
X#endif
X    char *p;
X    DIR  *dirp;
X    struct direct *d;
X    char *cur,*arc;
X    extern struct stat hstat;        /* Stat struct corresponding */
X    long size,copied;
X    char *from,*to;
X
X    dirp=opendir(name);
X
X    if(!dirp) {
X              /* The directory doesn't exist now.  It'll be created.
X                 In any case, we don't have to delete any files out
X                 of it */
X         return;
X    }
X
X    the_buffer=init_buffer();
X    while(d=readdir(dirp)) {
X         if(is_dot_or_dotdot(d->d_name))
X              continue;
X
X         add_buffer(the_buffer,d->d_name,(int)(DP_NAMELEN(d)+1));
X    }
X    closedir(dirp);
X    add_buffer(the_buffer,"",1);
X
X    current_dir=get_buffer(the_buffer);
X    archive_dir=(char *)malloc(hstat.st_size);
X    if(archive_dir==0) {
X         fprintf(stderr,"Can't allocate %d bytes for restore\n",hstat.st_size);
X         return;
X    }
X    to=archive_dir;
X    for(size=hstat.st_size;size>0;size-=copied) {
X         from=findrec()->charptr;
X         if(!from) {
```

```
X             msg("Unexpected EOF in archive\n");
X             break;
X        }
X        copied=endofrecs()->charptr - from;
X        if(copied>size)
X             copied=size;
X        bcopy((void *)from,(void *)to,(int)copied);
X        to+=copied;
X        userec((union record *)(from+copied-1));
X    }
X
X    for(cur=current_dir;*cur;cur+=strlen(cur)+1) {
X        for(arc=archive_dir;*arc;arc+=strlen(arc)+1) {
X             arc++;
X             if(!strcmp(arc,cur))
X                  break;
X        }
X        if(*arc=='\0') {
X             p=new_name(name,cur);
X             if(f_confirm && !confirm("delete",p)) {
X                  free(p);
X                  continue;
X             }
X             if(f_verbose)
X                  printf("%s: deleting %s\n",p);
X             if(recursively_delete(p)) {
X                  msg("%s: Error while deleting\n",p);
X             }
X             free(p);
X        }
X
X    }
X    flush_buffer(the_buffer);
X    free(archive_dir);
X}
X
Xrecursively_delete(path)
Xchar *path;
X{
X    struct stat sbuf;
X    DIR *dirp;
X    struct direct *dp;
X    char *path_buf;
X    /* int path_len; */
X
X
X    if(lstat(path,&sbuf)<0)
X        return 1;
X    if((sbuf.st_mode &S_IFMT)==S_IFDIR) {
X
X        /* path_len=strlen(path); */
X        dirp=opendir(path);
X        if(dirp==0)
```

```
X             return 1;
X        while(dp=readdir(dirp)) {
X             if(is_dot_or_dotdot(dp->d_name))
X                  continue;
X             path_buf=new_name(path,dp->d_name);
X             if(recursively_delete(path_buf)) {
X                  free(path_buf);
X                  closedir(dirp);
X                  return 1;
X             }
X             free(path_buf);
X        }
X        closedir(dirp);
X
X        if(rmdir(path)<0)
X             return 1;
X        return 0;
X    }
X    if(unlink(path)<0)
X         return 1;
X    return 0;
X}
X
Xchar *
Xnew_name(path,name)
Xchar *path,*name;
X{
X    char *path_buf;
X
X    path_buf=(char *)malloc(strlen(path)+strlen(name)+2);
X    if(path_buf==0) {
X         fprintf(stderr,"Can't allocate memory for name '%s/%s\n",path,name);
X         exit(EX_SYSTEM);
X    }
X    (void) sprintf(path_buf,"%s/%s",path,name);
X    return path_buf;
X}
X
X/* return non-zero if p is the name of a directory */
Xis_a_directory(p)
Xchar *p;
X{
X    struct stat stbuf;
X
X    if(lstat(p,&stbuf)<0) {
X         msg_perror("can't stat %s",p);
X         return 0;
X    }
X    if((stbuf.st_mode&S_IFMT)==S_IFDIR)
X         return 1;
X    return 0;
X}
X
```

```
X/* Returns non-zero if p is . or ..   This could be a macro for speed. */
Xis_dot_or_dotdot(p)
Xchar *p;
X{
X    return (p[0]=='.' && (p[1]=='\0' || (p[1]=='.' && p[2]=='\0')));
X}
X
X/* returns non-zero if the luser typed 'y' or 'Y', zero otherwise. */
X
Xint
Xconfirm(action,file)
Xchar *action, *file;
X{
X    int  c,nl;
X    static FILE *confirm_file = 0;
X    extern FILE *msg_file;
X
X    fprintf(msg_file,"%s %s?", action, file);
X    if(!confirm_file) {
X#ifdef atarist
X        confirm_file = (archive == 0) ? fopen("CON:", 0) : stdin;
X#else
X        confirm_file = (archive == 0) ? fopen("/dev/tty") : stdin;
X#endif
X        if(!confirm_file) {
X            fprintf(stderr,"Can't read confirmation from user\n");
X            exit(EX_SYSTEM);
X        }
X    }
X    c=getc(confirm_file);
X    for(nl = c; nl != '\n' && nl != EOF; nl = getc(confirm_file))
X        ;
X    return (c=='y' || c=='Y');
X}
X
Xchar *x_buffer = 0;
Xint size_x_buffer;
Xint free_x_buffer;
X
Xchar **exclude = 0;
Xint size_exclude = 0;
Xint free_exclude = 0;
X
Xchar **re_exclude = 0;
Xint size_re_exclude = 0;
Xint free_re_exclude = 0;
X
Xadd_exclude(file)
Xchar *file;
X{
X    FILE *fp;
X    char buf[1024];
X    extern char *rindex();
```

```
X
X    fp=fopen(file,"r");
X    if(!fp) {
X         msg_perror("can't open %s",file);
X         exit(2);
X    }
X    while(fgets(buf,1024,fp)) {
X         int size_buf;
X         char *end_str;
X
X         end_str=rindex(buf,'\n');
X         *end_str='\0';
X
X         un_quote_string(buf);
X         size_buf = strlen(buf);
X
X         if(x_buffer==0) {
X              x_buffer = (char *)ck_malloc(size_buf+1024);
X              free_x_buffer=1024;
X         } else if(free_x_buffer<=size_buf) {
X              char *old_x_buffer;
X              char **tmp_ptr;
X
X              old_x_buffer = x_buffer;
X              x_buffer = (char *)ck_realloc(x_buffer,size_x_buffer+1024);
X              free_x_buffer = 1024;
X              for(tmp_ptr=exclude;tmp_ptr<exclude+size_exclude;tmp_ptr++)
X                   *tmp_ptr= x_buffer + ((*tmp_ptr) - old_x_buffer);
X              for(tmp_ptr=re_exclude;tmp_ptr<re_exclude+size_re_exclude;tmp_ptr++)
X                   *tmp_ptr= x_buffer + ((*tmp_ptr) - old_x_buffer);
X         }
X
X         if(is_regex(buf)) {
X              if(free_re_exclude==0) {
X                   re_exclude= (char **)(re_exclude ? ck_realloc(re_exclude,
(size_re_exclude+32)*sizeof(char *)) : ck_malloc(sizeof(char *)*32));
X                   free_re_exclude+=32;
X              }
X              re_exclude[size_re_exclude]=x_buffer+size_x_buffer;
X              size_re_exclude++;
X              free_re_exclude--;
X         } else {
X              if(free_exclude==0) {
X                   exclude=(char **)(exclude ? ck_realloc(exclude,
(size_exclude+32)*sizeof(char *)) : ck_malloc(sizeof(char *)*32));
X                   free_exclude+=32;
X              }
X              exclude[size_exclude]=x_buffer+size_x_buffer;
X              size_exclude++;
X              free_exclude--;
X         }
X         strcpy(x_buffer+size_x_buffer,buf);
X         size_x_buffer+=size_buf+1;
```

```
X        free_x_buffer-=size_buf+1;
X    }
X    fclose(fp);
X}
X
Xint
Xis_regex(str)
Xchar *str;
X{
X    return index(str,'*') || index(str,'[') || index(str,'?');
X}
X
Xint
Xcheck_exclude(name)
Xchar *name;
X{
X    int n;
X
X    for(n=0;n<size_re_exclude;n++) {
X        if(wildmat(name,re_exclude[n]))
X            return 1;
X    }
X    for(n=0;n<size_exclude;n++) {
X        if(strstr(name,exclude[n]))
X            return 1;
X    }
X    return 0;
X}
SHAR_EOF
echo "File TAR.C is complete"
chmod 0644 TAR.C || echo "restore of TAR.C fails"
echo "x - extracting TAR.H (Text)"
sed 's/^X//' << 'SHAR_EOF' > TAR.H &&
X/*
X
X    Copyright (C) 1988 Free Software Foundation
X
XGNU tar is distributed in the hope that it will be useful, but WITHOUT ANY
XWARRANTY.  No author or distributor accepts responsibility to anyone
Xfor the consequences of using it or for whether it serves any
Xparticular purpose or works at all, unless he says so in writing.
XRefer to the GNU tar General Public License for full details.
X
XEveryone is granted permission to copy, modify and redistribute GNU tar,
Xbut only under the conditions described in the GNU tar General Public
XLicense.  A copy of this license is supposed to have been given to you
Xalong with GNU tar so you can know your rights and responsibilities.  It
Xshould be in a file named COPYING.  Among other things, the copyright
Xnotice and this notice must be preserved on all copies.
X
XIn other words, go ahead and share GNU tar, but don't try to stop
Xanyone else from sharing it farther.  Help stamp out software hoarding!
X*/
```

```
X
X/*
X * Header file for tar (tape archive) program.
X *
X * @(#)tar.h 1.24 87/11/06
X *
X * Created 25 August 1985 by John Gilmore, ihnp4!hoptoad!gnu.
X */
X
X/*
X * Kludge for handling systems that can't cope with multiple
X * external definitions of a variable.  In ONE routine (tar.c),
X * we #define TAR_EXTERN to null; here, we set it to "extern" if
X * it is not already set.
X */
X#ifndef TAR_EXTERN
X#define TAR_EXTERN extern
X#endif
X
X/*
X * Header block on tape.
X *
X * I'm going to use traditional DP naming conventions here.
X * A "block" is a big chunk of stuff that we do I/O on.
X * A "record" is a piece of info that we care about.
X * Typically many "record"s fit into a "block".
X */
X#define  RECORDSIZE    512
X#define  NAMSIZ   100
X#define  TUNMLEN  32
X#define  TGNMLEN  32
X
Xunion record {
X    char     charptr[RECORDSIZE];
X    struct header {
X         char name[NAMSIZ];
X         char mode[8];
X         char uid[8];
X         char gid[8];
X         char size[12];
X         char mtime[12];
X         char chksum[8];
X         char linkflag;
X         char linkname[NAMSIZ];
X         char magic[8];
X         char uname[TUNMLEN];
X         char gname[TGNMLEN];
X         char devmajor[8];
X         char devminor[8];
X         /* these following fields were added by JF for gnu */
X         /* and are NOT standard */
X         char atime[12];
X         char ctime[12];
```

```
X         char offset[12];
X         char longnames[4];
X    } header;
X};
X
X/* The checksum field is filled with this while the checksum is computed. */
X#define  CHKBLANKS"        "    /* 8 blanks, no null */
X
X/* The magic field is filled with this if uname and gname are valid. */
X#define  TMAGIC        "ustar  " /* 7 chars and a null */
X
X/* The linkflag defines the type of file */
X#define  LF_OLDNORMAL  '\0'     /* Normal disk file, Unix compat */
X#define  LF_NORMAL '0'       /* Normal disk file */
X#define  LF_LINK       '1'      /* Link to previously dumped file */
X#define  LF_SYMLINK    '2'      /* Symbolic link */
X#define  LF_CHR        '3'      /* Character special file */
X#define  LF_BLK        '4'      /* Block special file */
X#define  LF_DIR        '5'      /* Directory */
X#define  LF_FIFO       '6'      /* FIFO special file */
X#define  LF_CONTIG '7'       /* Contiguous file */
X/* Further link types may be defined later. */
X
X#define LF_DUMPDIR    'D'      /* This is a dir entry that contains
X                          the names of files that were in
X                          the dir at the time the dump
X                          was made */
X#define LF_MULTIVOL   'M'      /* This is the continuation
X                          of a file that began on another
X                          volume */
X#define LF_VOLHDR 'V'       /* This file is a tape/volume header */
X                       /* Ignore it on extraction */
X
X/*
X * Exit codes from the "tar" program
X */
X#define  EX_SUCCESS    0        /* success! */
X#define  EX_ARGSBAD    1        /* invalid args */
X#define  EX_BADFILE    2        /* invalid filename */
X#define  EX_BADARCH    3        /* bad archive */
X#define  EX_SYSTEM 4         /* system gave unexpected error */
X
X
X/*
X * Global variables
X */
XTAR_EXTERN union record    *ar_block;    /* Start of block of archive */
XTAR_EXTERN union record    *ar_record;   /* Current record of archive */
XTAR_EXTERN union record    *ar_last; /* Last+1 record of archive block */
XTAR_EXTERN char       ar_reading;   /* 0 writing, !0 reading archive */
XTAR_EXTERN int        blocking; /* Size of each block, in records */
XTAR_EXTERN int        blocksize;    /* Size of each block, in bytes */
XTAR_EXTERN char       *ar_file; /* File containing archive */
```

```
XTAR_EXTERN char       *name_file;   /* File containing names to work on */
XTAR_EXTERN char       *tar;         /* Name of this program */
X
X/*
X * Flags from the command line
X */
XTAR_EXTERN int cmd_mode;
X#define CMD_NONE  0
X#define CMD_CAT       1        /* -A */
X#define CMD_CREATE    2        /* -c */
X#define CMD_DIFF  3        /* -d */
X#define CMD_APPEND    4        /* -r */
X#define CMD_LIST  5        /* -t */
X#define CMD_UPDATE    6        /* -u */
X#define CMD_EXTRACT   7        /* -x */
X#define CMD_DELETE    8        /* -D */
X
X                     /* -[0-9][lmh] */
X             /* CMD_CAT       -A */
X                     /* -b */
XTAR_EXTERN char   f_reblock;         /* -B */
X             /* CMD_CREATE    -c */
X                     /* -C */
X             /* CMD_DIFF      -d */
X/* TAR_EXTERN char    f_dironly;    /* -D */
X                     /* -f */
XTAR_EXTERN char f_gnudump;      /* -G */
XTAR_EXTERN char   f_follow_links;        /* -h */
XTAR_EXTERN char   f_ignorez;         /* -i */
X             /* CMD_DELETE    -J */
XTAR_EXTERN char   f_keep;            /* -k */
XTAR_EXTERN char f_startfile;         /* -K */
XTAR_EXTERN char f_local_filesys;     /* -l */
XTAR_EXTERN char   f_modified;        /* -m */
XTAR_EXTERN char f_multivol;          /* -M */
XTAR_EXTERN char f_new_files;         /* -N */
XTAR_EXTERN char   f_oldarch;         /* -o */
XTAR_EXTERN char f_exstdout;          /* -O */
XTAR_EXTERN char   f_use_protection;  /* -p */
XTAR_EXTERN char   f_sayblock;        /* -R */
XTAR_EXTERN char   f_sorted_names;        /* -s */
XTAR_EXTERN char   f_namefile;        /* -T */
X             /* CMD_UPDATE    -u */
XTAR_EXTERN char   f_verbose;         /* -v */
XTAR_EXTERN char *f_volhdr;      /* -V */
XTAR_EXTERN char f_confirm;      /* -w */
XTAR_EXTERN char f_verify;       /* -W */
X             /* CMD_EXTRACT     -x */
XTAR_EXTERN char f_exclude;      /* -X */
XTAR_EXTERN char   f_compress;        /* -z */
X                     /* -Z */
X
X/*
```

```
X * We now default to Unix Standard format rather than 4.2BSD tar format.
X * The code can actually produce all three:
X *  f_standard    ANSI standard
X *  f_oldarch V7
X *  neither       4.2BSD
X * but we don't bother, since 4.2BSD can read ANSI standard format anyway.
X * The only advantage to the "neither" option is that we can cmp(1) our
X * output to the output of 4.2BSD tar, for debugging.
X */
X#define      f_standard        (!f_oldarch)
X
X/*
X * Structure for keeping track of filenames and lists thereof.
X */
Xstruct name {
X    struct name   *next;
X    short         length;       /* cached strlen(name) */
X    char     found;        /* A matching file has been found */
X    char     firstch;  /* First char is literally matched */
X    char     regexp;       /* This name is a regexp, not literal */
X    char     *change_dir;  /* JF set with the -C option */
X    char     *dir_contents;     /* JF for f_gnudump */
X    char     name[NAMSIZ+1];
X};
X
XTAR_EXTERN struct name *namelist;    /* Points to first name in list */
XTAR_EXTERN struct name *namelast;    /* Points to last name in list */
X
XTAR_EXTERN int        archive;  /* File descriptor for archive file */
XTAR_EXTERN int        errors;       /* # of files in error */
X
X/*
X *
X * Due to the next struct declaration, each routine that includes
X * "tar.h" must also include <sys/types.h>.  I tried to make it automatic,
X * but System V has no defines in <sys/types.h>, so there is no way of
X * knowing when it has been included.  In addition, it cannot be included
X * twice, but must be included exactly once.  Argghh!
X *
X * Thanks, typedef.  Thanks, USG.
X */
Xstruct link {
X    struct link   *next;
X    dev_t         dev;
X    ino_t         ino;
X    short         linkcount;
X    char     name[NAMSIZ+1];
X};
X
XTAR_EXTERN struct link *linklist;    /* Points to first link in list */
X
X
X/*
```

```
X * Error recovery stuff
X */
XTAR_EXTERN char       read_error_flag;
X
X
X/*
X * Declarations of functions available to the world.
X */
Xunion record *findrec();
Xvoid userec();
Xunion record *endofrecs();
Xvoid anno();
Xvoid msg();
Xvoid msg_perror();
X/* #define    annorec(stream, msg)  anno(stream, msg, 0)
     /* Cur rec */
X/* #define   annofile(stream, msg)  anno(stream, msg, 1)
     /* Saved rec */
SHAR_EOF
chmod 0644 TAR.H || echo "restore of TAR.H fails"
echo "x - extracting TAR.TEX (Text)"
sed 's/^X//' << 'SHAR_EOF' > TAR.TEX &&
X\input texinfo @c -*-texinfo-*-
X
X@setfilename tar
X@settitle tar
X@ifinfo
XThis file documents the tape archive of the GNU system.
X
XCopyright (C) 1988 Free Software Foundation, Inc.
X
XPermission is granted to make and distribute verbatim copies of this
Xmanual provided the copyright notice and this permission notice are
Xpreserved on all copies.
X
X@ignore
XPermission is granted to process this file through TeX and print the
Xresults, provided the printed document carries copying permission
Xnotice identical to this one except for the removal of this paragraph
X(this paragraph not being relevant to the printed manual).
X
X@end ignore
XPermission is granted to copy and distribute modified versions of
Xthis manual under the conditions for verbatim copying, provided that
Xthe entire resulting derived work is distributed under the terms of
Xa permission notice identical to this one.
X
XPermission is granted to copy and distribute translations of this
Xmanual into another language, under the above conditions for
Xmodified versions.
X@end ifinfo
X
X@iftex
```

```
X@finalout
X@end iftex
X
X@titlepage
X@center @titlefont{tar}
X@sp 1
X@center The GNU tape archive
X@sp 2
X@center Jay Fenlason
X@sp 3
XThis manual describes the GNU tape archiver, @code{tar}, and how you
Xcan use it to store copies of a file or a group of files in an
X@dfn{archive}.  This archive may be written directly to a magnetic
Xtape or other storage medium, stored as a file, or sent through a
Xpipe to another program.  @code{tar} can also be used to add files
Xto an already existing archive, list the files in an archive, or
Xextract the files in the archive.
X
XGNU @code{tar} was written by John Gilmore, and modified by many
Xpeople.  The GNU enhancements were written by Jay Fenlason.
X@page
X@vskip 0pt plus 1filll
XCopyright @copyright{} 1988 Free Software Foundation, Inc.
X
XPermission is granted to make and distribute verbatim copies of
Xthis manual provided the copyright notice and this permission notice
Xare preserved on all copies.
X
X@ignore
XPermission is granted to process this file through Tex and print the
Xresults, provided the printed document carries copying permission
Xnotice identical to this one except for the removal of this paragraph
X(this paragraph not being relevant to the printed manual).
X
X@end ignore
XPermission is granted to copy and distribute modified versions of this
Xmanual under the conditions for verbatim copying, provided that the entire
Xresulting derived work is distributed under the terms of a permission
Xnotice identical to this one.
X
XPermission is granted to copy and distribute translations of this manual
Xinto another language, under the same conditions as for modified versions.
X
X@end titlepage
X
X@ifinfo
X@node Top, Why, , (dir)
X@ichapter Using the Tape Archiver
X@menu
X* Why::           What @code{tar} archives are good for.
X* Commands::      How to tell @code{tar} what to do.
X* Options::       Options that change the way @code{tar} behaves.
X* Format::        The format of a @code{tar} archive.
```

```
X@end menu
X@end ifinfo
X
X@node Why, Commands, Top, Top
X@chapter The Uses of Tape Archives
X
XThe tape archiver @code{tar} allows you to store many files in an
X@dfn{archive file} or @dfn{tar file} which describes the names and contents
Xof the constituent files.  Later you can extract some or all of these files
Xfrom the archive.
X
XTar files are not restricted to magnetic tapes.  The @code{tar} program
Xcan equally well use an ordinary file, or a pipe, or any device, as the
Xarchive.  But they were originally designed for use with magnetic tapes,
Xand that is how the name ``tar'' came about.
X
XArchive files can be used for transporting a group of files from one system
Xto another:  put all relevant files into an archive on one computer system,
Xtransfer the archive to another, and extract the contents there.  The basic
Xtransfer medium might be magnetic tape, Internet FTP, or even electronic
Xmail (though you must encode the archive with @code{uuencode} in order to
Xtransport it properly by mail).  Both machines do not have to use the same
Xoperating system, as long as they both support the @code{tar} program.
X
XA magnetic tape can store several files in sequence, but has no names for
Xthem, just relative position on the tape.  A tar file or something like it
Xis the only way to store several files on one tape and retain their names.
XEven when the basic transfer mechanism can keep track of names, as FTP can,
Xthe nuisance of handling multiple files, directories, and multiple links,
Xmay make a tar file a much easier method.
X
XArchive files are also used for long-term storage, which you can think
Xof as transportation from one time to another.
X
XPiping one @code{tar} to another is an easy way to copy a directory's
Xcontents from one disk to another, while preserving the dates, modes, owners
Xand link-structure of all the files therein.
X
XThe GNU version of @code{tar} has special features that allow it to be
Xused to make incremental and full dumps of all the files in a
Xfilesystem.
X
X@node Commands, Options, Why, Top
X@chapter The Different Operations @code{tar} Can Perform
X
XOne program, @code{tar}, is used to create an archive, to extract files
Xfrom an archive, to modify an archive, or to list the contents.  Each
Xtime you run @code{tar}, you must give a @dfn{command} to specify which
Xone of these things you want to do.
X
XThe command must always be in the first argument to @code{tar}.  This
Xargument can also contain options (@pxref{Options}).  For compatibility
Xwith Unix @code{tar}, the first argument is always treated as containing
```

```
Xcommand and option letters even if it doesn't start with @samp{-}.  Thus,
X@samp{tar c} is equivalent to @samp{tar -c}: both of them specify the
X@samp{-c} command to create an archive.
X
XThe remaining arguments to @code{tar} are either options, if they start
Xwith @samp{-}, or files to operate on.
X
XThe file names that you give as argument are the files that @code{tar} will
Xact on--for example, they are the files to put in the archive, or the files
Xto extract from it.  If you don't give any file name arguments, the default
Xdepends on which command you used.  Some commands use all relevant files;
Xsome commands have no default and will report an error if you don't specify
Xfiles.
X
XIf a file name argument actually names a directory, then that directory
Xand all files and subdirectories in it are used.
X
XHere is a list of all the @code{tar} commands:
X
X@table @samp
X@item -c
XThe @samp{-c} command tells @code{tar} to create a new archive that
Xcontains the file(s) specified on the command line.  If you don't
Xspecify files, all the files in the current directory are used.
X
XIf the archive file already exists, it is overwritten; the old contents
Xare lost.
X
X@item -d
XThe @samp{-d} command causes @code{tar} to compare the archive with
Xthe files in the file system.  It will report differences in file
Xsize, mode, owner, and contents.  If a file exists in the archive, but
Xnot in the file system, @code{tar} will report this.
X
XIf you specify file names, those files are compared with the tape and
Xthey must all exist in the archive.  If you don't specify files, all
Xthe files in the archive are compared.
X
X@item -r
XThe @samp{-r} command causes @code{tar} to add the specified file(s)
Xto the end of the archive.  This assumes that the archive file already
Xexists and is in the proper format (which probably means it was
Xcreated previously with the @code{tar} program).  If the archive is
Xnot in a format that @code{tar} understands, the results will be
Xunpredictable.
X
XYou must specify the files to be used; there is no default.
X
X@item -t
XThe @samp{-t} command causes @code{tar} to display a list of the files
Xin the archive.  If you specify file names, only the files
Xthat you specify will be mentioned (but each of them is mentioned only
Xif it appears in the archive).
```

```
X
X@item -u
XThe @samp{-u} command causes @code{tar} to add the specified files to
Xthe end of the archive, like @w{@samp{-r}}, but only when a file doesn't
Xalready exist in the archive or is newer than the version in the
Xarchive (last-modification time is compared).  This command can be
Xvery slow.
X
XYou must specify the files to be used; there is no default.
X
X@item -x
XThe @samp{-x} command causes @code{tar} to extract the specified files
Xfrom the archive.  If no file names are given, all the files in the
Xarchive will be extracted.
X
X@item -A
XThe @samp{-A} command is used for concatenating several archive files
Xinto one big archive file.  The files to operate on should all be
Xarchive files.  They are all appended to the end of @emph{the} archive
Xfile which @code{tar} works on.  (The other files are not changed).
X
XYou might be tempted to use @code{cat} for this, but it won't
Xordinarily work.  A @code{tar} archive contains data which indicates
Xthe end of the archive, so more material added to the end with
X@code{cat} would be ignored.  The @samp{tar -A} command works because
Xit removes the end-of-archive markers from the middle of the result.
X
X@item -D
XThe @samp{-D} command causes @code{tar} to delete the specified files
Xfrom the archive.  This command is extremely slow.  Warning:  Use of
Xthis command on archives stored on magnetic tape may result in a
Xscrambled archive.  There is no safe way (except for completely
Xre-writing the archive) to delete files from a magnetic tape.
X@end table
X
X@node Options, General Options, Commands, Top
X@chapter Options That Change How @code{tar} Works
X
XOptions may be specified as individual arguments starting with @samp{-}.
XIn this case, if the option wants an argument (as does, for example,
X@samp{-f}) then the argument should come after the option, separated
Xfrom it by a space.
XAll options are optional.  Some options make sense with any command, while
Xothers are meaningful only with particular commands.@refill
X
X@menu
X* General Options::
X* Creation Options::
X* Extraction Options::
X* Option Syntax::
X@end menu
X
X@node General Options, Creation Options, Options, Options
```

```
X@section Options That Are Always Meaningful
X
X@table @code
X@item -b @var{number}
XThis option is used to specify a @dfn{blocking factor} for the
Xarchive.  When reading or writing the archive, @code{tar}, will do
Xreads and writes of the archive in blocks of @var{number}*512 bytes.
X
XThe default blocking factor is set when @code{tar} is compiled, and is
Xtypically 20.
X
XBlocking factors larger than 20 cannot be read by very old versions of
X@code{tar}, or by some newer versions of @code{tar} running on old machines
Xwith small address spaces.
X
XWith a magnetic tape, larger blocks give faster throughput and fit
Xmore data on a tape (because there are fewer inter-record gaps).  If
Xthe archive is in a disk file or a pipe, you may want to specify a
Xsmaller blocking factor, since a large one will result in a large
Xnumber of null bytes at the end of the archive.
X
XWhen writing cartridge or other streaming tapes, a much larger
Xblocking factor (say 126 or more) will greatly increase performance.
XHowever, you must specify the same blocking factor when reading or
Xupdating the archive.
X
XWith GNU @code{tar} the blocking factor is limited only by the maximum
Xblock size of the device containing the archive, or by the amount of
Xavailable virtual memory.
X
X@item -f @var{filename}
XThis option is used to specify the file name of the archive @code{tar}
Xworks on.
X
XIf no @samp{-f} option is given, but the environment variable
X@code{TAPE} exists, its value is used; otherwise, a default archive
Xname (which was picked when @code{tar} was compiled) is used.  The
Xdefault is normally set up to be the ``first'' tape drive or other
Xtransportable I/O medium on the system.
X
XIf the filename is @samp{-}, @code{tar} reads the archive from
Xstandard input (when listing or extracting), or writes it to standard
Xoutput (when creating).  If the @samp{-} filename is given when
Xupdating an archive, @code{tar} will read the original archive from
Xits standard input, and will write the entire new archive to its
Xstandard output.
X
XIf the filename contains @samp{:/dev/}, it is interpreted as
X@samp{hostname:filename}.  If the @var{hostname} contains an ampersand
X(@samp{@@}), it is treated as @samp{user@@hostname:filename}.
XIn either case, @code{tar} will invoke the command @code{rsh}
X(or @code{remsh}) to start
Xup an @code{/etc/rmt} on the remote machine.  If you give an alternate login
```

```
Xname, it will be given to the @code{rsh}.  Naturally, the remote machine must
Xhave a copy of @file{/etc/rmt}.  @code{/etc/rmt} is free software
Xfrom the University of California, and a copy of the source code can be found
Xwith the sources for @code{tar}.  @code{/etc/rmt} will have to be modified to
Xrun on non-BSD4.3 systems.@refill
SHAR_EOF
echo "End of part 6"
echo "File TAR.TEX is continued in part 7"
echo "7" > s2_seq_.tmp
exit 0


From 5089479948060914554
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,d19a44957f4a36eb,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-19 14:43:41 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
decwrl!sun-barr!newstop!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i053: gnutar -- GNU's version of UNIX "tar"  part07/08
Keywords: shar, C
Message-ID: <33893@grapevine.uucp>
Date: 19 Jul 89 21:43:41 GMT
Sender: news@grapevine.uucp
Lines: 1226
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: 7103_300@uwovax.uwo.ca (Eric R. Smith)
Posting-number: Volume 2, Issue 53
Archive-name: gnutar/part07

#!/bin/sh
# this is part 7 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file TAR.TEX continued
#
CurArch=7
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file TAR.TEX"
sed 's/^X//' << 'SHAR_EOF' >> TAR.TEX
X
X@item -C @var{dir}
XThe @samp{-C} option causes @code{tar} to change into the the
```

```
Xdirectory @var{dir} before continuing.  This option is usually
Xinterspersed with the files @code{tar} is to work on.  For example,
X
X@example
Xtar -c iggy ziggy -C baz melvin
X@end example
X
X@noindent
Xwill place the files @file{iggy} and @file{ziggy} from the current
Xdirectory on the tape, followed by the file @file{melvin} from the
Xdirectory @file{baz}.  This option is especially useful when you have
Xseveral widely separated files that you want to store in the same
Xdirectory in the archive.
X
XNote that the file @file{melvin} is recorded in the archive under the
Xprecise name @file{melvin}, @emph{not} @file{baz/melvin}.  Thus, the
Xarchive will contain three files that all appear to have come from the
Xsame directory; if the archive is extracted with plain @samp{tar -x},
Xall three files will be created in the current directory.
X
XContrast this with the command
X
X@example
Xtar -c iggy ziggy bar/melvin
X@end example
X
X@noindent
Xwhich records the third file in the archive under the name @file{bar/melvin}
Xso that, if plain @samp{tar -x} is used, the third file will be created
Xin a subdirectory named @file{bar}.
X
X@item -M
XThe @samp{-M} option causes @code{tar} to write a @dfn{multi-volume}
Xarchive--one that may be larger than will fit on the medium used to
Xhold it.
X
XWhen this option is used, @code{tar} will not abort when it cannot
Xread or write any more data.  Instead, it will ask you to prepare a
Xnew volume.  If the archive is on a magnetic tape, you should change
Xtapes now; if the archive is on a floppy disk, you should change
Xdisks, etc.
X
XEach volume of a multi-volume archive is an independent tar archive,
Xcomplete in itself.  For example, you can list or extract any volume
Xalone (just don't specify @samp{-M}).  However, if one file in the
Xarchive is split across volumes, the only way to extract it
Xsuccessfully is with a multi-volume extract command (@samp{-xM})
Xstarting on or before the volume where the file begins.
X
X@item -N @var{date}
XThis option causes @code{tar} to only work on files whose modification
Xor inode-changed times are newer than the @var{date} given.  The main
Xuse is for creating an archive; then only new files are written.  If
```

```
Xextracting, only newer files are extracted.
X
XRemember that the entire date argument should be quoted if it contains
Xany spaces.
X
XThe date is parsed using @code{getdate}.
X
X@item -R
XIf @samp{-R} is used, @code{tar} prints, along with every message it
Xwould normally produce, the record number within the archive where
Xthe message occurred.  This option is especially useful when reading
Xdamaged archives, since it helps pinpoint the damaged sections.
X
XThis can also be useful when making a log of a file-system backup tape,
Xsince the results allow you to find the file you want to retrieve
Xon several backup tapes and choose the tape where the file appears
Xearliest (closest to the front of the tape).
X
X@item -T @var{filename}
XInstead of taking the list of files to work on from the command
Xline, the list of files to work on is read from the file
X@var{filename}.  If @var{filename} is given as @samp{-}, the list is
Xread from standard input.  Note that using both @samp{-T -} and
X@samp{-f -} will not work unless you are using the @samp{-c} command.
X
X@item -v
XThis option causes @code{tar} to be verbose about the actions it is
Xtaking.
X
XNormally, the @samp{-t} command to list an archive prints
Xjust the file names (one per line) and the other commands are silent.
X
X@samp{-tv} prints a full line of information about each file, like the
Xoutput of @samp{ls -l}.  @samp{-v} with any other command (aside from
X@samp{-t}) prints just the name of each file operated on.
X
XThe output from @samp{-v} appears on the standard output except when
Xcreating or updating an archive to the standard output,
Xin which case the output from @samp{-v} is sent to the standard
Xerror.
X
X@item -version
X
XThis option causes @code{tar} to print out its version number to the
Xstandard error.  In order to avoid being confused with the @samp{-v}
Xoption, @samp{-version} must be given as a separate option,
Xpreceded by a hyphen.
X
X@item -w
XThis option causes @code{tar} to print a message for each action it
Xintends to take, and ask for confirmation on the terminal.  To
Xconfirm, you must type a line of input.  If your input line begins
Xwith @samp{y}, the action is performed, otherwise it is skipped.
```

```
X
XThe actions which require confirmation include adding a file to the
Xarchive, extracting a file from the archive, deleting a file from the
Xarchive, and deleting a file from disk.
X
XIf @code{tar} is reading the archive from the standard input,
X@code{tar} will open the file @file{/dev/tty} to ask for
Xconfirmation on.
X
X@item -X @var{file}
XThis option causes @code{tar} to read a list of filenames
X(actually regular expressions) from the file @var{file};
X@code{tar} will ignore files with those names.
XThus if @code{tar} is called as @samp{tar -c -X foo .}
Xand the file @file{foo} contains @samp{*.o} none of the files
Xwhose names end in @file{.o} in the current directory will be added
Xto the archive.  Multiple @code{-X} options may be given.
X
X@item -z
X@itemx -Z
XThe archive should be compressed as it is written, or decompressed
Xas it is read, using the @code{compress} program.  This option works
Xon physical devices (tape drives, etc.) and remote files as well as
Xon normal files; data to or from such devices or remote files is
Xreblocked by another copy of the @code{tar} program to enforce the
Xspecified (or default) block size.  The default compression
Xparameters are used; if you need to override them, avoid the
X@samp{-z} option and run @code{compress} explicitly.
X
XNote that the @samp{-z} option will not work with the @samp{-M} option,
Xor with the @samp{-u}, @samp{-r}, @samp{-A}, or @samp{-D} commands.
X@end table
X
X@node Creation Options, Extraction Options, General Options, Options
X@section Options for Creating Or Updating an Archive
X
XThese options are used to control which files @code{tar} puts in an
Xarchive, or to control the format the archive is written in (@pxref{Format}).
X
XExcept as noted below, these options are useful with the @samp{-c},
X@samp{-r}, @samp{-u}, @samp{-A}, and @samp{-D} commands.
XAlso note that the @samp{-B} option, (@pxref{Extraction Options}),
Xis also useful with the @samp{-r}, @samp{-u}, @samp{-A}, and @samp{-D} commands.
X
X@table @code
X@c this command no longer exists  -D is now the old -J command
X@c @item -D
X@c The @samp{-D} option tells @code{tar} to only store entries for the
X@c directories it encounters, and to not to store the files inside the
X@c directories.  In conjunction with @code{find} this is useful for
X@c creating incremental dumps for archival backups, similar to those
X@c produced by @code{dump}.
X
```

```
X@item -G
XThis option should only be used when creating an incremental backup of
Xa filesystem.  When the @samp{-G} option is used, @code{tar} writes, at
Xthe beginning of the archive, an entry for each of the directories that
Xwill be operated on.  The entry for a directory includes a list of all
Xthe files in the directory at the time the dump was done, and a flag
Xfor each file indicating whether the file is going to be put in the
Xarchive.  This information is used when doing a complete incremental
Xrestore.
X
XNote that this option causes @code{tar} to create a non-standard
Xarchive that may not be readable by non-GNU versions of the @code{tar}
Xprogram.
X
X@item -h
XIf @samp{-h} is used, when @code{tar} encounters a symbolic link, it
Xwill archive the linked-to file, instead of simply recording the
Xpresence of a symbolic link.  If the linked-to file is archived
Xagain, an entire second copy of it will be archived, instead of a
Xlink.  This could be considered a bug.
X
X@item -l
XThis option causes @code{tar} to not cross filesystem boundaries
Xwhen archiving parts of a directory tree.  This option only
Xaffects files that are archived because they are in a directory that
Xis archived; files named on the command line are archived
Xregardless, and they can be from various file systems.
X
XThis option is useful for making full or incremental archival backups of
Xa file system, as with the Unix @code{dump} command.
X
XFiles which are skipped due to this option are mentioned on the
Xstandard error.
X
X@item -o
XThis option causes @code{tar} to write an old format archive, which
Xdoes not include information about directories, pipes, fifos,
Xcontiguous files, or device files, and specifies file ownership by
Xnumeric user- and group-ids rather than by user and group names.  In
Xmost cases, a @emph{new} format archive can be read by an @emph{old}
X@code{tar} program without serious trouble, so this option should
Xseldom be needed.  When updating an archive, do not use @samp{-o}
Xunless the archive was created with the @samp{-o} option.
X
X@item -V @var{name}
XThis option causes @code{tar} to write out a @dfn{volume header} at
Xthe beginning of the archive.  If @samp{-M} is used, each volume of
Xthe archive will have a volume header of @samp{@var{name} Volume @var{N}},
Xwhere @var{N} is 1 for the first volume, 2 for the next, and so on.
X
X@item -W
XThis option causes @code{tar} to verify the archive after writing it.
XEach volume is checked after it is written, and any discrepancies are
```

```
Xrecorded on the standard error output.
X
XVerification requires that the archive be on a back-space-able medium.
XThis means pipes, some cartridge tape drives, and some other devices
Xcannot be verified.
X@end table
X
X@node Extraction Options, Option Syntax, Creation Options, Options
X@section Options for Listing Or Extracting Files
X
XThe options in this section are meaningful with the @samp{-x} command.
XUnless otherwise stated, they are also meaningful with the @samp{-t}
Xcommand.
X
X@table @code
X@item -B
XIf @samp{-B} is used, @code{tar} will not panic if an attempt to
Xread a block from the archive does not return a full block.  Instead,
X@code{tar} will keep reading until it has obtained a full block.
X
XThis option is turned on by default when @code{tar} is reading an
Xarchive from standard input, or from a remote machine.  This is
Xbecause on BSD Unix systems a read of a pipe will return however much
Xhappens to be in the pipe, even if it is less than @code{tar}
Xrequested.  If this option was not used, @code{tar} would fail
Xas soon as it read an incomplete block from the pipe.
X
XThis option is also useful with the commands for updating an archive.
X
X@item -G
XThe @samp{-G} option means the archive is an incremental backup.
XIts meaning depends on the command that it modifies.
X
XIf the @samp{-G} option is used with @samp{-t}, @code{tar} will list,
Xfor each directory in the archive, the list of files in that
Xdirectory at the time the archive was created.  This information is
Xput out in a format that is not easy for humans to read, but which
Xis unambiguous for a program: each filename is preceded by either a
X@samp{Y} if the file is present in the archive, or an @samp{N} if the
Xfile is a directory, or is not included in the archive.  Each
Xfilename is terminated by a null character.  The last file is
Xfollowed by an additional null and a newline to indicate the end of
Xthe data.
X
XIf the @samp{-G} option is used with @samp{-x}, then when the entry
Xfor a directory is found, all files that currently exist in that directory
Xbut are not listed in the archive @emph{are deleted from the directory}.
X
XThis behavior is convenient when you are restoring a damaged file system
Xfrom a succession of incremental backups: it restores the entire state
Xof the file system to that which obtained when the backup was made.
XIf you don't use @samp{-G}, the file system will probably fill up
Xwith files that shouldn't exist any more.
```

```
X
X@item -i
XThe @samp{-i} option causes @code{tar} to ignore blocks of zeros in the
Xarchive.  Normally a block of zeros indicates the end of the
Xarchive, but when reading a damaged archive, or one which was created by
X@code{cat}-ing several archives together, this option allows
X@code{tar} to read the entire archive.  This option is not on by
Xdefault because many versions of @code{tar} write garbage after the
Xzeroed blocks.
X
XNote that this option causes @code{tar} to read to the end of the
Xarchive file, which may sometimes avoid problems when multiple files
Xare stored on a single physical tape.
X
X@item -k
XThe @samp{-k} option prevents @code{tar} from over-writing existing
Xfiles with files with the same name from the archive.
X
XThe @samp{-k} option is meaningless with @samp{-t}.
X
X@item -K @var{filename}
XThe @samp{-K} option causes @code{tar} to begin extracting or listing
Xthe archive with the file @var{filename}, and to consider only the
Xfiles starting at that point in the archive.  This is useful if a
Xprevious attempt to extract files failed when it reached
X@var{filename} due to lack of free space.  (This assumes, of course,
Xthat there is now free space, or that you are now extracting into a
Xdifferent file system.)
X
X@item -m
XWhen this option is used, @code{tar} leaves the modification times of
Xthe files it extracts as the time when the files were extracted,
Xinstead of setting it to the time recorded in the archive.
X
XThe @samp{-m} option is meaningless with @samp{-t}.
X
X@item -O
XWhen this option is used, instead of creating the files
Xspecified, @code{tar} writes the contents of the files
Xextracted to its standard output.  This may be useful if you
Xare only extracting the files in order to send them through a
Xpipeline.
X
XThe @samp{-O} option is meaningless with @samp{-t}.
X
X@item -p
XThis option causes @code{tar} to set the modes (access permissions) of
Xextracted files exactly as recorded in the archive.  If this option is
Xnot used, the current @code{umask} setting limits the permissions on
Xextracted files.
X
XThe @samp{-p} option is meaningless with @samp{-t}.
X
```

```
X@item -s
XThe @samp{-s} option tells @code{tar} that the list of filenames to be
Xlisted or extracted is sorted in the same order as the files in the
Xarchive.  This allows a large list of names to be used, even on a
Xsmall machine that would not otherwise be able to hold all the names
Xin memory at the same time.  Such a sorted list can easily be created
Xby running @samp{tar -t} on the archive and editing its output.
X
X@samp{-s} is probably never needed on modern computer systems.
X@end table
X
X@node Option Syntax,, Extraction Options, Options
X@section Old Syntax for Options
X
XFor compatibility with Unix @code{tar}, the first argument can contain
Xoption letters in addition to the command letter; for example, @samp{tar
Xcv} specifies the option @samp{-v} in addition to the command @samp{-c}.
XThe first argument to GNU @code{tar} is always treated as command and
Xoption letters even if it doesn't start with @samp{-}.
X
XSome options need their own arguments; for example, @samp{-f} is followed
Xby the name of the archive file.  When the option is given separately, its
Xargument follows it, as is usual for Unix programs.  For example:
X
X@example
Xtar -c -v -b 20 -f /dev/rmt0
X@end example
X
XWhen options that need arguments are given together with the command, all
Xthe associated arguments follow, in the same order as the options.  Thus,
Xthe example above could also be written in the old style as follows:
X
X@example
Xtar cvbf 20 /dev/rmt0
X@end example
X
X@noindent
XHere @samp{20} is the argument of @samp{-b} and @file{/dev/rmt0} is the
Xargument of @samp{-f}.
X
X@node FullDumps, Extraction Options, IncDumps, Top
X@chapter Using @code{tar} to Perform Full Dumps
XFull dumps should only be made when no other people or programs are
Xmodifying files in the filesystem.  If files are modified while
X@code{tar} is making the backup, they may not be stored properly in
Xthe archive, in which case you won't be able to restore them if you
Xhave to.
X
XYou will want to use the @samp{-V} option to give the archive a
Xvolume label, so you can tell what this archive is even if the label
Xfalls off the tape, or anything like that.
X
XUnless the filesystem you are dumping is guaranteed to fit on one
```

```
Xvolume, you will need to use the @samp{-M} option.  Make sure you
Xhave enough tapes on hand to complete the backup.
X
XIf you want to dump each filesystem separately you will need to use
Xthe @samp{-l} option to prevent @code{tar} from crossing filesystem
Xboundaries when storing (sub)directories.
X
XThe @samp{-G} option is not needed, since this is a complete copy of
Xeverything in the filesystem, and a full restore from this backup
Xwould only be done onto a completely empty disk.
X
XUnless you are in a hurry, and trust the @code{tar} program (and
Xyour tapes), it is a good idea to use the @code{-W} (verify) option,
Xto make sure your files really made it onto the dump properly.  This
Xwill also detect cases where the file was modified while (or just
Xafter) it was being archived.
X
X@node IncDumps, FullDumps, Problems, Top
X@chapter Using @code{tar} to Perform Incremental Dumps
XPerforming incremental dumps is similar to performing full dumps,
Xalthough a few more options will usually be needed.
X
XYou will need to use the @samp{-N @var{date}} option to tell @code{tar} to
Xonly store files that have been modified since @var{date}.
X@var{date} should be the date and time of the last full/incremental
Xdump.
X
XA standard scheme is to do a @samp{monthly} (full) dump once a month,
Xa @samp{weekly} dump once a week of everything since the last monthly and
Xa @samp{daily} every day of everything since the last (weekly or monthly)
Xdump.
X
XHere is a copy of the script used to dump the filesystems of the
Xmachines here at the Free Software Foundation.  This script is run
X(semi-)automatically late at night when people are least likely to
Xbe using the machines.  This script dumps several filesystems from
Xseveral machines at once (by using a network-filesystem).  The
Xoperator is responsible for ensuring that all the machines will be
Xup at the time the dump happens.  If a machine is not running, its
Xfiles will not be dumped, and the next day's incremental dump will
X@emph{not} store files that would have gone onto that dump.
X
X@example
X#!/bin/csh
X# Dump thingie
Xset now = `date`
Xset then = `cat date.nfs.dump`
X/u/hack/bin/tar -c -G -v\
X -f /dev/rtu20\
X -b 126\
X -N "$then"\
X -V "Dump from $then to $now"\
X /alpha-bits/gp\
```

```
X /gnu/hack\
X /hobbes/u\
X /spiff/u\
X /sugar-bombs/u
Xecho $now > date.nfs.dump
Xmt -f /dev/rtu20 rew
X@end example
X
XOutput from this script is stored in a file, for the operator to
Xread later.
X
XThis script uses the file @file{date.nfs.dump} to store the date/time of
Xthe last dump.
X
XSince this is a streaming tape drive, no attempt to verify the
Xarchive is done.  This is also why the high blocking factor (126) is
Xused.  The tape drive must also be rewound by the @code{mt} command
Xafter the dump is made.
X
X@node Problems, IncDumps, Format, Top
X@chapter Common Problems Using @code{tar}
X
XIn order to update an archive, @code{tar} must be able to backspace
Xthe archive in order to re-read or re-write a block that was just read
X(or written).  This is currently possible only on two kinds of
Xfiles:  normal disk files (or any other file that can be
Xbackspaced with @code{lseek()}), and industry-standard 9-track magnetic
Xtape (or any other kind of tape that can be backspaced with
X@code{ioctl(@dots{},MTIOCTOP,@dots{})}).
X
XThis means that the @samp{-r}, @samp{-u}, @samp{-A}, and @samp{-D}
Xcommands will not work on any other kind of file.  Some media simply
Xcannot be backspaced, which means these commands and options will
Xnever be able to work on them.  These non-backspacing media include
Xpipes and cartridge tape drives.
X
XSome other media can be backspaced, and @code{tar} will work on them
Xonce it is modified to do so.
X
XArchives created with the @samp{-M}, @samp{-V}, and @samp{-G}
Xoptions may not be readable by other version of @code{tar}.  In particular,
Xrestoring a file that was split over a volume boundary will require
Xsome careful work with @code{dd}, if it can be done at all.  Other versions
Xof @code{tar} may also create an empty file whose name is that of
Xthe volume header.  Some versions of @code{tar} may create normal
Xfiles instead of directories archived with the @samp{-G} option.
X
X@node Rem Tape, IncDumps, Format, Top
X@chapter The Remote Tape Server
XIn order to access the tape drive on a remote machine, @code{tar}
Xuses the remote tape server written at the University of California
Xat Berkeley.  The remote tape server must be installed as
X@file{/etc/rmt} on any machine whose tape drive you want to use.
```

```
X@code{tar} calls @file{/etc/rmt} by running an @code{rsh} or
X@code{remsh} to the remote machine, optionally using a different
Xlogin name if one is supplied.
X
XA copy of the source for the remote tape server is provided.  It is
XCopyright @copyright 1983 by the Regents of the University of California, but
Xcan be freely distributed.  Instructions for compiling and
Xinstalling it are included in the @file{Makefile}.
X
XThe remote tape server may need to be modified in order to run on a
Xnon-4.3BSD system.
X
X@node Format, Rem Tape, , Top
X@chapter The Format of a @code{tar} Archive
XThis chapter is based heavily on John Gilmore's @i{tar}(5) manual page
Xfor the public domain @code{tar} that GNU @code{tar} is based on.
X
X@section The Standard Format
XA @dfn{tar tape} or file contains a series of records.  Each record
Xcontains @code{RECORDSIZE} bytes.  Although this format may be
Xthought of as being on magnetic tape, other media are often used.
X
XEach file archived is represented by a header record which describes
Xthe file, followed by zero or more records which give the contents
Xof the file.  At the end of the archive file there may be a record
Xfilled with binary zeros as an end-of-file marker.  A reasonable
Xsystem should write a record of zeros at the end, but must not
Xassume that such a record exists when reading an archive.
X
XThe records may be @dfn{blocked} for physical I/O operations.  Each
Xblock of @var{N} records (where @var{N} is set by the @samp{-b}
Xoption to @code{tar}) is written with a single @code{write()}
Xoperation.  On magnetic tapes, the result of such a write is a
Xsingle tape record.  When writing an archive, the last block of
Xrecords should be written at the full size, with records after the
Xzero record containing all zeroes.  When reading an archive, a
Xreasonable system should properly handle an archive whose last block
Xis shorter than the rest, or which contains garbage records after a
Xzero record.
X
XThe header record is defined in C as follows:
X
X@example
X/*
X * Standard Archive Format - Standard TAR - USTAR
X */
X#define  RECORDSIZE  512
X#define  NAMSIZ      100
X#define  TUNMLEN      32
X#define  TGNMLEN      32
X
Xunion record @{
X    char        charptr[RECORDSIZE];
```

```
X    struct header @{
X        char    name[NAMSIZ];
X        char    mode[8];
X        char    uid[8];
X        char    gid[8];
X        char    size[12];
X        char    mtime[12];
X        char    chksum[8];
X        char    linkflag;
X        char    linkname[NAMSIZ];
X        char    magic[8];
X        char    uname[TUNMLEN];
X        char    gname[TGNMLEN];
X        char    devmajor[8];
X        char    devminor[8];
X    @} header;
X@};
X
X/* The checksum field is filled with this while the checksum is computed. */
X#define    CHKBLANKS    "        "        /* 8 blanks, no null */
X
X/* The magic field is filled with this if uname and gname are valid. */
X#define    TMAGIC    "ustar  "        /* 7 chars and a null */
X
X/* The magic field is filled with this if this is a GNU format dump entry */
X#define    GNUMAGIC  "GNUtar "        /* 7 chars and a null */
X
X/* The linkflag defines the type of file */
X#define  LF_OLDNORMAL '\0'       /* Normal disk file, Unix compatible */
X#define  LF_NORMAL    '0'        /* Normal disk file */
X#define  LF_LINK      '1'        /* Link to previously dumped file */
X#define  LF_SYMLINK   '2'        /* Symbolic link */
X#define  LF_CHR       '3'        /* Character special file */
X#define  LF_BLK       '4'        /* Block special file */
X#define  LF_DIR       '5'        /* Directory */
X#define  LF_FIFO      '6'        /* FIFO special file */
X#define  LF_CONTIG    '7'        /* Contiguous file */
X
X/* Further link types may be defined later. */
X
X/* Bits used in the mode field - values in octal */
X#define  TSUID    04000        /* Set UID on execution */
X#define  TSGID    02000        /* Set GID on execution */
X#define  TSVTX    01000        /* Save text (sticky bit) */
X
X/* File permissions */
X#define  TUREAD   00400        /* read by owner */
X#define  TUWRITE  00200        /* write by owner */
X#define  TUEXEC   00100        /* execute/search by owner */
X#define  TGREAD   00040        /* read by group */
X#define  TGWRITE  00020        /* write by group */
X#define  TGEXEC   00010        /* execute/search by group */
X#define  TOREAD   00004        /* read by other */
```

```
X#define  TOWRITE  00002        /* write by other */
X#define  TOEXEC   00001        /* execute/search by other */
X@end example
X
XAll characters in header records are represented by using 8-bit
Xcharacters in the local variant of ASCII.  Each field within the
Xstructure is contiguous; that is, there is no padding used within
Xthe structure.  Each character on the archive medium is stored
Xcontiguously.
X
XBytes representing the contents of files (after the header record of
Xeach file) are not translated in any way and are not constrained to
Xrepresent characters in any character set.  The @code{tar} format
Xdoes not distinguish text files from binary files, and no
Xtranslation of file contents is performed.
X
XThe @code{name}, @code{linkname}, @code{magic}, @code{uname}, and
X@code{gname} are null-terminated character strings.  All other
Xfileds are zero-filled octal numbers in ASCII.  Each numeric field
Xof width @var{w} contains @var{w}@minus 2 digits, a space, and a null,
Xexcept @code{size}, and @code{mtime}, which do not contain the
Xtrailing null.
X
XThe @code{name} field is the pathname of the file, with directory
Xnames (if any) preceding the file name, separated by slashes.
X
XThe @code{mode} field provides nine bits specifying file permissions
Xand three bits to specify the Set UID, Set GID, and Save Text
X(``stick'') modes.  Values for these bits are defined above.  When
Xspecial permissions are required to create a file with a given mode,
Xand the user restoring files from the archive does not hold such
Xpermissions, the mode bit(s) specifying those special permissions
Xare ignored.  Modes which are not supported by the operating system
Xrestoring files from the archive will be ignored.  Unsupported modes
Xshould be faked up when creating or updating an archive; e.g. the
Xgroup permission could be copied from the @code{other} permission.
X
XThe @code{uid} and @code{gid} fields are the numeric user and group
XID of the file owners, respectively.  If the operating system does
Xnot support numeric user or group IDs, these fields should be
Xignored.
X
XThe @code{size} field is the size of the file in bytes; linked files
Xare archived with this field specified as zero.
X@xref{Extraction Options}; in particular the @samp{-G} option.@refill
X
XThe @code{mtime} field is the modification time of the file at the
Xtime it was archived.  It is the ASCII representation of the octal
Xvalue of the last time the file was modified, represented as an
Xinteger number of seconds since January 1, 1970, 00:00 Coordinated
XUniversal Time.
X
XThe @code{chksum} field is the ASCII representation of the octal
```

```
Xvalue of the simple sum of all bytes in the header record.  Each
X8-bit byte in the header is added to an unsigned integer,
Xinitialized to zero, the precision of which shall be no less than
Xseventeen bits.  When calculating the checksum, the @code{chksum}
Xfield is treated as if it were all blanks.
X
XThe @code{typeflag} field specifies the type of file archived.  If a
Xparticular implementation does not recognize or permit the specified
Xtype, the file will be extracted as if it were a regular file.  As
Xthis action occurs, @code{tar} issues a warning to the standard
Xerror.
X
X@table @code
X@item LF_NORMAL
X@itemx LF_OLDNORMAL
XThese represent a regular file.  In order to be compatible with
Xolder versions of @code{tar}, a @code{typeflag} value of
X@code{LF_OLDNORMAL} should be silently recognized as a regular
Xfile.  New archives should be created using @code{LF_NORMAL}.  Also,
Xfor backward compatibility, @code{tar} treats a regular file whose
Xname ends with a slash as a directory.
X
X@item LF_LINK
XThis represents a file linked to another file, of any type,
Xpreviously archived.  Such files are identified in Unix by each file
Xhaving the same device and inode number.  The linked-to
Xname is specified in the @code{linkname} field with a trailing null.
X
X@item LF_SYMLINK
XThis represents a symbolic link to another file.  The linked-to
Xname is specified in the @code{linkname} field with a trailing null.
X
X@item LF_CHR
X@itemx LF_BLK
XThese represent character special files and block special files
Xrespectively.  In this case the @code{devmajor} and @code{devminor}
Xfields will contain the major and minor device numbers
Xrespectively.  Operating systems may map the device specifications
Xto their own local specification, or may ignore the entry.
X
X@item LF_DIR
XThis specifies a directory or sub-directory.  The directory name in
Xthe @code{name} field should end with a slash.  On systems where
Xdisk allocation is performed on a directory basis the @code{size}
Xfield will contain the maximum number of bytes (which may be rounded
Xto the nearest disk block allocation unit) which the directory may
Xhold.  A @code{size} field of zero indicates no such limiting.
XSystems which do not support limiting in this manner should ignore
Xthe @code{size} field.
X
X@item LF_FIFO
XThis specifies a FIFO special file.  Note that the archiving of a
XFIFO file archives the existence of this file and not its contents.
```

```
X
X@item LF_CONTIG
XThis specifies a contiguous file, which is the same as a normal
Xfile except that, in operating systems which support it,
Xall its space is allocated contiguously on the disk.  Operating
Xsystems which do not allow contiguous allocation should silently treat
Xthis type as a normal file.
X
X@item 'A' @dots{}
X@itemx 'Z'
XThese are reserved for custom implementations.  Some of these are
Xused in the GNU modified format, as described below.
X@end table
X
XOther values are reserved for specification in future revisions of
Xthe P1003 standard, and should not be used by any @code{tar} program.
X
XThe @code{magic} field indicates that this archive was output in the
XP1003 archive format.  If this field contains @code{TMAGIC}, the
X@code{uname} and @code{gname} fields will contain the ASCII
Xrepresentation of the owner and group of the file respectively.  If
Xfound, the user and group ID represented by these names will be used
Xrather than the values within the @code{uid} and @code{gid} fields.
X
X@section GNU Extensions to the Archive Format
XThe GNU format uses additional file types to describe new types of
Xfiles in an archive.  These are listed below.
X
X@table @code
X@item LF_DUMPDIR
X@itemx 'D'
XThis represents a directory and a list of files created by the
X@samp{-G} option.  The @code{size} field gives the total size of the
Xassociated list of files.  Each filename is preceded by either a @code{'Y'}
X(the file should be in this archive) or an @code{'N'} (The file is a
Xdirectory, or is not stored in the archive).  Each filename is
Xterminated by a null.  There is an additional null after the last
Xfilename.
X
X@item LF_MULTIVOL
X@itemx 'M'
XThis represents a file continued from another volume of a
Xmulti-volume archive created with the @samp{-M} option.  The original
Xtype of the file is not given here.  The @code{size} field gives the
Xmaximum size of this piece of the file (assuming the volume does not
Xend before the file is written out).  The @code{offset} field gives
Xthe offset from the beginning of the file where this part of the
Xfile begins.  Thus @code{size} plus @code{offset} should equal the
Xoriginal size of the file.
X
X@item LF_VOLHDR
X@itemx 'V'
XThis file type is used to mark the volume header that was given with
```

```
Xthe @samp{-V} option when the archive was created.  The @code{name}
Xfield contains the @code{name} given after the @samp{-V} option.
XThe @code{size} field is zero.  Only the first file in each volume
Xof an archive should have this type.
X
X@end table
X
XYou may have trouble reading a GNU format archive on a non-GNU system
Xif the options @samp{-G}, @samp{-M} or @samp{-V} were used when writing
Xthe archive.
X@unnumbered Concept Index
X@printindex cp
X@contents
X@bye
SHAR_EOF
echo "File TAR.TEX is complete"
chmod 0644 TAR.TEX || echo "restore of TAR.TEX fails"
echo "x - extracting UPDATE.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > UPDATE.C &&
X/*
X
X    Copyright (C) 1988 Free Software Foundation
X
XGNU tar is distributed in the hope that it will be useful, but WITHOUT ANY
XWARRANTY.  No author or distributor accepts responsibility to anyone
Xfor the consequences of using it or for whether it serves any
Xparticular purpose or works at all, unless he says so in writing.
XRefer to the GNU tar General Public License for full details.
X
XEveryone is granted permission to copy, modify and redistribute GNU tar,
Xbut only under the conditions described in the GNU tar General Public
XLicense.  A copy of this license is supposed to have been given to you
Xalong with GNU tar so you can know your rights and responsibilities.  It
Xshould be in a file named COPYING.  Among other things, the copyright
Xnotice and this notice must be preserved on all copies.
X
XIn other words, go ahead and share GNU tar, but don't try to stop
Xanyone else from sharing it farther.  Help stamp out software hoarding!
X*/
X
X/* JF implement the 'r' 'u' and 'A' options for tar. */
X/* The 'A' option is my own invention:  It means that the file-names are
X   tar files, and they should simply be appended to the end of the archive.
X   No attempt is made to block the reads from the args; if they're on raw
X   tape or something like that, it'll probably lose. . . */
X
X#ifdef atarist
X#include <types.h>
X#include <stat.h>
X#else
X#include <sys/types.h>
X#include <sys/stat.h>
X#endif
```

```
X#include <stdio.h>
X#include <errno.h>
X
X/* JF these includes are copied from create.c  I'm not sure if they're right
X   or not.  */
X#ifndef V7
X#include <fcntl.h>
X#endif
X
X#ifndef MSDOS
X#include <pwd.h>
X#include <grp.h>
X#endif
X
X#ifdef USG
X#include <sys/sysmacros.h> /* major() and minor() defined here */
X#endif
X
X/*
X * V7 doesn't have a #define for this.
X */
X#ifndef O_RDONLY
X#define O_RDONLY 0
X#endif
X
X/*
X * Most people don't have a #define for this.
X */
X#ifndef O_BINARY
X#define O_BINARY 0
X#endif
X
X#define STDIN 0
X#define STDOUT 1
X
X#include "tar.h"
X#include "port.h"
X#include "rmt.h"
X
Xint time_to_start_writing = 0;  /* We've hit the end of the old stuff,
X                     and its time to start writing new stuff
X                     to the tape.  This involves seeking
X                     back one block and re-writing the current
X                     block (which has been changed). */
X
Xchar *output_start;        /* Pointer to where we started to write in
X                     the first block we write out.  This is used
X                     if we can't backspace the output and have
X                     to null out the first part of the block */
X
Xextern void skip_file();
X
Xextern union record *head;
```

```
Xextern struct stat hstat;
X
Xstruct name *name_scan();
Xchar    *name_from_list();
X
X/* Implement the 'r' (add files to end of archive), and 'u' (add files to
X   end of archive if they arent there, or are more up to date than the
X   version in the archive.) commands.*/
Xvoid
Xupdate_archive()
X{
X    int  found_end = 0;
X    int  status = 3;
X    int  prev_status;
X    char *p;
X    struct name *name;
X    extern void dump_file();
X
X    name_gather();
X    if(cmd_mode==CMD_UPDATE)
X         name_expand();
X    open_archive(2);   /* Open for updating */
X
X    do {
X         prev_status=status;
X         status=read_header();
X         switch(status) {
X         case EOF:
X              found_end=1;
X              break;
X
X         case 0:       /* A bad record */
X              userec(head);
X              switch(prev_status) {
X              case 3:
X                   msg("This doesn't look like a tar archive.\n");
X                   /* FALL THROUGH */
X              case 2:
X              case 1:
X                   msg("Skipping to next header\n");
X              case 0:
X                   break;
X              }
X              break;
X
X              /* A good record */
X         case 1:
X /* printf("File %s\n",head->header.name); */
X              head->header.name[NAMSIZ-1]='\0';
X              if(cmd_mode==CMD_UPDATE && (name=name_scan(head->header.name))) {
X                   /* struct stat hstat; */
X                   struct stat nstat;
X                   int head_standard;
```

```
X
X                  decode_header(head,&hstat,&head_standard,0);
X                  if(stat(head->header.name,&nstat)<0) {
X                       msg_perror("can't stat %s:",head->header.name);
X                  } else {
X                       if(hstat.st_mtime>=nstat.st_mtime)
X                            name->found++;
X                  }
X             }
X             userec(head);
X             skip_file((long)hstat.st_size);
X             break;
X
X        case 2:
X             ar_record=head;
X             found_end = 1;
X             break;
X        }
X    } while(!found_end);
X
X    time_to_start_writing = 1;
X    output_start=ar_record->charptr;
X
X    while(p=name_from_list()) {
X         if(f_confirm && !confirm("add", p))
X              continue;
X         if(cmd_mode==CMD_CAT)
X              append_file(p);
X         else
X              dump_file(p,-1);
X    }
X
X    write_eot();
X    close_archive();
X    names_notfound();
X}
X
X/* Catenate file p to the archive without creating a header for it.  It had
X   better be a tar file or the archive is screwed */
X
Xappend_file(p)
Xchar *p;
X{
X    int  fd;
X    struct stat statbuf;
X    long bytes_left;
X    union record *start;
X    long bufsiz,count;
X
X    if(0 != stat(p,&statbuf) || (fd=open(p,O_RDONLY|O_BINARY))<0) {
X         msg_perror("can't open file %s",p);
X         errors++;
X         return;
```

```
X    }
X
X    bytes_left = statbuf.st_size;
X
X    while(bytes_left>0) {
X         start=findrec();
X         bufsiz=endofrecs()->charptr - start->charptr;
X         if(bytes_left < bufsiz) {
X              bufsiz = bytes_left;
X              count = bufsiz % RECORDSIZE;
X              if(count)
X                   bzero(start->charptr + bytes_left,(int)(RECORDSIZE-count));
X         }
X         count=read(fd,start->charptr,bufsiz);
X         if(count<0) {
X              msg_perror("read error at byte %ld reading %d bytes in file
%s",statbuf.st_size-bytes_left,bufsiz,p);
X              exit(EX_ARGSBAD);      /* FOO */
X         }
X         bytes_left-=count;
X         userec(start+(count-1)/RECORDSIZE);
X         if(count!=bufsiz) {
X              msg("%s: file shrunk by %d bytes, yark!\n",p,bytes_left);
X              abort();
X         }
X    }
X    (void)close(fd);
X}
X
X#ifdef DONTDEF
Xbprint(fp,buf,num)
XFILE *fp;
Xchar *buf;
X{
X    int  c;
X
X    if(num==0 || num==-1)
X         return;
X    fputs(" '",fp);
X    while(num--) {
X         c= *buf++;
X         if(c=='\\') fputs("\\\\",fp);
X         else if(c>=' ' && c<='~')
X              putc(c,fp);
X         else switch(c) {
X         case '\n':
X              fputs("\\n",fp);
X              break;
X         case '\r':
X              fputs("\\r",fp);
X              break;
X         case '\b':
X              fputs("\\b",fp);
```

```
X             break;
X        case '\0':
X             /* fputs("\\-",fp); */
X             break;
X        default:
X             fprintf(fp,"\\%03o",c);
X             break;
X        }
X    }
X    fputs("'\n",fp);
X}
X#endif
X
Xint number_of_blocks_read = 0;
X
Xint number_of_new_records = 0;
Xint number_of_records_needed = 0;
X
Xunion record *new_block = 0;
Xunion record *save_block = 0;
X
Xvoid
Xjunk_archive()
X{
X    int  found_stuff = 0;
X    int  status = 3;
X    int  prev_status;
X    struct name *name;
X
X    /* int dummy_head; */
X    int number_of_records_to_skip = 0;
X    int number_of_records_to_keep = 0;
X    int number_of_kept_records_in_block;
X    int sub_status;
X    extern int write_archive_to_stdout;
X
X/* fprintf(stderr,"Junk files\n"); */
X    name_gather();
X    open_archive(2);
X
X    while(!found_stuff) {
X         prev_status=status;
X         status=read_header();
X         switch(status) {
X         case EOF:
X              found_stuff = 1;
X              break;
X
X         case 0:
X              userec(head);
X              switch(prev_status) {
X              case 3:
X                   msg("This doesn't look like a tar archive.\n");
```

```
X                 /* FALL THROUGH */
X             case 2:
X             case 1:
X                 msg("Skipping to next header\n");
X                 /* FALL THROUGH */
X             case 0:
X                 break;
X             }
X             break;
X
X         case 1:
X             head->header.name[NAMSIZ-1] = '\0';
X /* fprintf(stderr,"file %s\n",head->header.name); */
X             if((name=name_scan(head->header.name))==(struct name *)0) {
X                 userec(head);
X /* fprintf(stderr,"Skip %ld\n",(long)(hstat.st_size)); */
X                 skip_file((long)(hstat.st_size));
X                 break;
X             }
X             name->found = 1;
X             found_stuff = 2;
X             break;
X
X         case 2:
X             found_stuff = 1;
X             break;
X         }
X    }
X /* fprintf(stderr,"Out of first loop\n"); */
X
X    if(found_stuff!=2) {
X         write_eot();
X         close_archive();
X         names_notfound();
X         return;
X    }
X
X    if(write_archive_to_stdout)
X         write_archive_to_stdout = 0;
X    new_block = (union record *)malloc(blocksize);
X    if(new_block==0) {
X         fprintf(stderr,"Can't allocate secondary block of %d bytes\n",blocksize);
X         exit(EX_SYSTEM);
X    }
X
X         /* Save away records before this one in this block */
X    number_of_new_records=ar_record-ar_block;
X    number_of_records_needed = blocking - number_of_new_records;
X    if(number_of_new_records)
X         bcopy((void *)ar_block,(void *)new_block,
(number_of_new_records)*RECORDSIZE);
X
X /* fprintf(stderr,"Saved %d recs, need %d
```

```
more\n",number_of_new_records,number_of_records_needed); */
X    userec(head);
X    skip_file((long)(hstat.st_size));
X    found_stuff=0;
X    /* goto flush_file; */
X
X    for(;;) {
X              /* Fill in a block */
X    /* another_file: */
X         if(ar_record==ar_last) {
X /* fprintf(stderr,"New block\n"); */
X              flush_archive();
X              number_of_blocks_read++;
X         }
X         sub_status = read_header();
X /* fprintf(stderr,"Header type %d\n",sub_status); */
X
X         if(sub_status==2 && f_ignorez) {
X              userec(head);
X              continue;
X         }
X         if(sub_status==EOF || sub_status==2) {
X              found_stuff = 1;
X
bzero(new_block[number_of_new_records].charptr,RECORDSIZE*number_of_records_needed
);
X              number_of_new_records+=number_of_records_needed;
X              number_of_records_needed = 0;
X              write_block(0);
X              break;
X         }
X
X         if(sub_status==0) {
X              fprintf(stderr,"Deleting non-header from archive.\n");
X              userec(head);
X              continue;
X         }
X
X         /* Found another header.  Yipee! */
X         head->header.name[NAMSIZ-1] = '\0';
X /* fprintf(stderr,"File %s ",head->header.name); */
X         if(name=name_scan(head->header.name)) {
X              name->found = 1;
X /* fprintf(stderr,"Flush it\n"); */
X         /* flush_file: */
X              /* decode_header(head,&hstat,&dummy_head,0); */
X              userec(head);
X              number_of_records_to_skip=(hstat.st_size+RECORDSIZE-1)/RECORDSIZE;
X /* fprintf(stderr,"Flushing %d recs from %s\n",number_of_records_to_skip,head-
>header.name); */
X
X              while(ar_last-ar_record<=number_of_records_to_skip) {
X
```

```
X /* fprintf(stderr,"Block: %d <= %d  ",ar_last-
ar_record,number_of_records_to_skip); */
X                  number_of_records_to_skip -= (ar_last - ar_record);
X                  flush_archive();
X                  number_of_blocks_read++;
X /* fprintf(stderr,"Block %d left\n",number_of_records_to_skip); */
X             }
X             ar_record+=number_of_records_to_skip;
X /* fprintf(stderr,"Final %d\n",number_of_records_to_skip); */
X             number_of_records_to_skip = 0;
X             continue;
X        }
X
X    /* copy_header: */
X         new_block[number_of_new_records]= *head;
X         number_of_new_records++;
X         number_of_records_needed--;
X         number_of_records_to_keep=(hstat.st_size+RECORDSIZE-1)/RECORDSIZE;
X         userec(head);
X         if(number_of_records_needed==0)
X              write_block(1);
X    /* copy_data: */
X         number_of_kept_records_in_block = ar_last - ar_record;
X         if(number_of_kept_records_in_block > number_of_records_to_keep)
X              number_of_kept_records_in_block = number_of_records_to_keep;
X
X /* fprintf(stderr,"Need %d kept_in %d keep
%d\n",blocking,number_of_kept_records_in_block,number_of_records_to_keep); */
X
X         while(number_of_records_to_keep) {
X              int n;
X
X              if(ar_record==ar_last) {
X /* fprintf(stderr,"Flush. . .\n"); */
X                   fl_read();
X                   number_of_blocks_read++;
X                   ar_record=ar_block;
X                   number_of_kept_records_in_block = blocking;
X                   if(number_of_kept_records_in_block > number_of_records_to_keep)
SHAR_EOF
echo "End of part 7"
echo "File UPDATE.C is continued in part 8"
echo "8" > s2_seq_.tmp
exit 0


From 6545189176874148630
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,24e2e0d63a7e41d1,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-19 14:43:49 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!ucbvax!
agate!apple!sun-barr!newstop!sun!grapevine!koreth%panarthea.ebay@sun.com
```

From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i054: gnutar -- GNU's version of UNIX "tar" part08/08
Keywords: shar, C
Message-ID: <33894@grapevine.uucp>
Date: 19 Jul 89 21:43:49 GMT
Sender: news@grapevine.uucp
Lines: 319
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: 7103_300@uwovax.uwo.ca (Eric R. Smith)
Posting-number: Volume 2, Issue 54
Archive-name: gnutar/part08

```
#!/bin/sh
# this is part 8 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file UPDATE.C continued
#
CurArch=8
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file UPDATE.C"
sed 's/^X//' << 'SHAR_EOF' >> UPDATE.C
X                       number_of_kept_records_in_block =
number_of_records_to_keep;
X             }
X             n = number_of_kept_records_in_block;
X             if(n>number_of_records_needed)
X                  n = number_of_records_needed;
X
X /* fprintf(stderr,"Copying %d\n",n); */
X             bcopy((void *)ar_record, (void *)(new_block+number_of_new_records),
n*RECORDSIZE);
X             number_of_new_records           += n;
X             number_of_records_needed        -= n;
X             ar_record                       += n;
X             number_of_records_to_keep       -= n;
X             number_of_kept_records_in_block -= n;
X /* fprintf(stderr,"Now new %d  need %d  keep %d  keep_in %d rec %d/%d\n",
X number_of_new_records,number_of_records_needed,number_of_records_to_keep,
X number_of_kept_records_in_block,ar_record-ar_block,ar_last-ar_block); */
X
X             if(number_of_records_needed == 0) {
X                  write_block(1);
X             }
```

```
X        }
X    }
X
X    write_eot();
X    close_archive();
X    names_notfound();
X}
X
Xwrite_block(f)
X{
X /* fprintf(stderr,"Write block\n"); */
X    /* We've filled out a block.  Write it out. */
X
X    /* Backspace back to where we started. . . */
X    if(archive!=STDIN)
X         (void)move_arch(-(number_of_blocks_read+1));
X
X    save_block = ar_block;
X    ar_block = new_block;
X
X    if(archive==STDIN)
X         archive=STDOUT;
X    fl_write();
X
X    if(archive==STDOUT)
X         archive=STDIN;
X    ar_block = save_block;
X
X    if(f) {
X         /* Move the tape head back to where we were */
X         if(archive!=STDIN)
X              (void)move_arch(number_of_blocks_read);
X         number_of_blocks_read--;
X    }
X
X    number_of_records_needed = blocking;
X    number_of_new_records = 0;
X}
X
X/* Move archive descriptor by n blocks worth.  If n is positive we move
X   forward, else we move negative.   If its a tape, MTIOCTOP had better
X   work.  If its something else, we try to seek on it.  If we can't
X   seek, we lose! */
Xmove_arch(n)
X{
X    long cur;
X    extern int errno;
X
X#ifdef MTIOCTOP
X    struct mtop t;
X    int er;
X
X    if(n>0) {
```

```
X        t.mt_op = MTFSR;
X        t.mt_count = n;
X    } else {
X        t.mt_op = MTBSR;
X        t.mt_count = -n;
X    }
X    if((er=rmtioctl(archive,MTIOCTOP,&t))>=0)
X        return 1;
X    if(errno==EIO && (er=rmtioctl(archive,MTIOCTOP,&t))>=0)
X        return 1;
X#endif
X
X    cur=rmtlseek(archive,0L,1);
X    cur+=blocksize*n;
X
X /* fprintf(stderr,"Fore to %x\n",cur); */
X    if(rmtlseek(archive,cur,0)!=cur) {
X        /* Lseek failed.  Try a different method */
X        fprintf(stderr,"tar: Couldn't re-position archive file.\n");
X        exit(EX_BADARCH);
X    }
X    return 3;
X}
X
SHAR_EOF
echo "File UPDATE.C is complete"
chmod 0644 UPDATE.C || echo "restore of UPDATE.C fails"
echo "x - extracting VERSION.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > VERSION.C &&
X#ifdef atarist
Xchar version_string[] = "GNU tar version 1.04 (GEMDOS)";
X#else
Xchar version_string[] = "GNU tar version 1.04";
X#endif
X/* Version 1.00:  This file added.  -version option added */
X
X/*      Installed new version of the remote-tape library */
X/*      Added -help option */
X/*  1.01 Fixed typoes in tar.texinfo */
X/*      Fixed a bug in the #define for rmtcreat() */
X/*      Fixed the -X option to not call realloc() of 0. */
X
X/*  1.02 Fixed tar.c so 'tar -h' and 'tar -v' don't cause core dump */
X/*      Also fixed the 'usage' message to be more up-to-date. */
X/*      Fixed diffarch.c so verify should compile without MTIOCTOP
X            defined */
X
X/*  1.03 Fixed buffer.c so 'tar tzf NON_EXISTENT_FILE' returns an error
X            message instead of hanging forever */
X
X/*      More fixes to tar.texinfo */
X/*  1.04 Added functions msg() and msg_perror()  Modified all the
X            files to call them.  Also checked that all (I hope)
```

```
X             calls to msg_perror() have a valid errno value
X             (modified anno() to leave errno alone), etc
X         Re-fixed the -X option.  This
X         time for sure. . . */
X/*       re-modified the msg stuff.  flushed anno() completely */
X/*       Modified the directory stuff so it should work on sysV boxes */
X/*       added ftime() to getdate.y */
X/*       Fixed un_quote_string() so it won't wedge on \" Also fixed
X         \ddd (like \123, etc) */
X/*       More fixes to tar.texinfo */
SHAR_EOF
chmod 0644 VERSION.C || echo "restore of VERSION.C fails"
echo "x - extracting WILDMAT.C (Text)"
sed 's/^X//' << 'SHAR_EOF' > WILDMAT.C &&
X/*
X
X    Copyright (C) 1988 Free Software Foundation
X
XGNU tar is distributed in the hope that it will be useful, but WITHOUT ANY
XWARRANTY.  No author or distributor accepts responsibility to anyone
Xfor the consequences of using it or for whether it serves any
Xparticular purpose or works at all, unless he says so in writing.
XRefer to the GNU tar General Public License for full details.
X
XEveryone is granted permission to copy, modify and redistribute GNU tar,
Xbut only under the conditions described in the GNU tar General Public
XLicense.  A copy of this license is supposed to have been given to you
Xalong with GNU tar so you can know your rights and responsibilities.  It
Xshould be in a file named COPYING.  Among other things, the copyright
Xnotice and this notice must be preserved on all copies.
X
XIn other words, go ahead and share GNU tar, but don't try to stop
Xanyone else from sharing it farther.  Help stamp out software hoarding!
X*/
X
X/*
X * @(#)wildmat.c 1.3 87/11/06
X *
XFrom: rs@mirror.TMC.COM (Rich Salz)
XNewsgroups: net.sources
XSubject: Small shell-style pattern matcher
XMessage-ID: <596@mirror.TMC.COM>
XDate: 27 Nov 86 00:06:40 GMT
X
XThere have been several regular-expression subroutines and one or two
Xfilename-globbing routines in mod.sources.  They handle lots of
Xcomplicated patterns.  This small piece of code handles the *?[]\
Xwildcard characters the way the standard Unix(tm) shells do, with the
Xaddition that "[^.....]" is an inverse character class -- it matches
Xany character not in the range ".....".  Read the comments for more
Xinfo.
X
XFor my application, I had first ripped off a copy of the "glob" routine
```

```
Xfrom within the find(1) source, but that code is bad news:  it recurses
Xon every character in the pattern.  I'm putting this replacement in the
Xpublic domain.  It's small, tight, and iterative.  Compile with -DTEST
Xto get a test driver.  After you're convinced it works, install in
Xwhatever way is appropriate for you.
X
XI would like to hear of bugs, but am not interested in additions; if I
Xwere, I'd use the code I mentioned above.
X*/
X/*
X**  Do shell-style pattern matching for ?, \, [], and * characters.
X**  Might not be robust in face of malformed patterns; e.g., "foo[a-"
X**  could cause a segmentation violation.
X**
X**  Written by Rich $alz, mirror!rs, Wed Nov 26 19:03:17 EST 1986.
X*/
X
X/*
X * Modified 6Nov87 by John Gilmore (hoptoad!gnu) to return a "match"
X * if the pattern is immediately followed by a "/", as well as \0.
X * This matches what "tar" does for matching whole subdirectories.
X *
X * The "*" code could be sped up by only recursing one level instead
X * of two for each trial pattern, perhaps, and not recursing at all
X * if a literal match of the next 2 chars would fail.
X */
X#define TRUE      1
X#define FALSE          0
X
X
Xstatic int
XStar(s, p)
X    register char *s;
X    register char *p;
X{
X    while (wildmat(s, p) == FALSE)
X    if (*++s == '\0')
X        return(FALSE);
X    return(TRUE);
X}
X
X
Xint
Xwildmat(s, p)
X    register char *s;
X    register char *p;
X{
X    register int   last;
X    register int   matched;
X    register int   reverse;
X
X    for ( ; *p; s++, p++)
X    switch (*p) {
```

```
X        case '\\':
X         /* Literal match with following character; fall through. */
X         p++;
X        default:
X         if (*s != *p)
X             return(FALSE);
X         continue;
X        case '?':
X         /* Match anything. */
X         if (*s == '\0')
X             return(FALSE);
X         continue;
X        case '*':
X         /* Trailing star matches everything. */
X         return(*++p ? Star(s, p) : TRUE);
X        case '[':
X         /* [^....] means inverse character class. */
X         if (reverse = p[1] == '^')
X             p++;
X         for (last = 0400, matched = FALSE; *++p && *p != ']'; last = *p)
X             /* This next line requires a good C compiler. */
X             if (*p == '-' ? *s <= *++p && *s >= last : *s == *p)
X              matched = TRUE;
X         if (matched == reverse)
X             return(FALSE);
X         continue;
X    }
X
X    /* For "tar" use, matches that end at a slash also work. --hoptoad!gnu */
X    return(*s == '\0' || *s == '/');
X}
X
X
X#ifdef  TEST
X#include <stdio.h>
X
Xextern char  *gets();
X
X
Xmain()
X{
X    char pattern[80];
X    char text[80];
X
X    while (TRUE) {
X    printf("Enter pattern:  ");
X    if (gets(pattern) == NULL)
X        break;
X    while (TRUE) {
X        printf("Enter text:  ");
X        if (gets(text) == NULL)
X         exit(0);
X        if (text[0] == '\0')
```

```
X         /* Blank line; go back and get a new pattern. */
X         break;
X        printf("      %d\n", wildmat(text, pattern));
X    }
X    }
X    exit(0);
X}
X#endif  /* TEST */
SHAR_EOF
chmod 0644 WILDMAT.C || echo "restore of WILDMAT.C fails"
rm -f s2_seq_.tmp
echo "You have unpacked the last part"
exit 0


From -3339203423596261573
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,66362b833796bad5,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-20 16:21:32 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!rutgers!
apple!sun-barr!newstop!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i055: coloremu -- Color emulator for mono screens
Keywords: 68000
Message-ID: <33903@grapevine.uucp>
Date: 20 Jul 89 23:21:32 GMT
Sender: news@grapevine.uucp
Lines: 231
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: f-leoe@ifi.uio.no (Lars-Erik 0sterud)
Posting-number: Volume 2, Issue 55
Archive-name: coloremu

[The binary will be posted to the binaries group.  The source is below.
 The submitter asks that someone try to speed this up if possible... -sg]

---- cut here ----
 MOVE.L 4(A7),A0
 MOVE.L #$8400,D6               ; 32K for the screen
 ADD.L $C(A0),D6                ; plus the usual space
 ADD.L $14(A0),D6
 ADD.L $1C(A0),D6
 MOVE.L D6,-(SP)                ; save length of program for later

 MOVE.W #4,-(SP)
 TRAP #14                       ; Get screen Resolution
 ADDQ.L #2,SP
 CMP.W #2,D0                    ; If monochrome then carry on
 BEQ monochrome
 MOVE.L (SP)+,D6                ; else tidy up the stack
```

```
 MOVE.W #0,-(SP)                ; return ok to,GEM
 TRAP #1                        ; Then exit back to desktop
monochrome:
 CLR.L -(SP)
 MOVE.W #32,-(SP)
 TRAP #1                        ; Enter supervisor mode
 ADDQ.L #6,SP
 MOVE.L D0,savestack            ; Save the supervisor stack

 PEA message                    ; Address of message
 MOVE.W #9,-(SP)                ; Print startup message
 TRAP #1
 ADDQ.L #6,SP
inloop:
 PEA input                      ; Adress of message
 MOVE.W #9,-(SP)                ; Print input message
 TRAP #1
 ADDQ.L #6,SP
 MOVE.W #1,-(SP)                ; Conin
 TRAP #1                        ; GemDos
 ADDQ.L #2,SP
 SUB.B #48,D0
 BLE inloop                     ; Too low  (<1)
 CMP.B #6,D0
 BGT inloop                     ; Too High (>6)
 MOVE.B D0,step                 ; Save speed
 SUB.L A5,A5
 MOVE.L $044E(A5),high          ; Set HIGH
 MOVE.L $B8,xbiospoke+2         ; Get the old XBIOS address and
 MOVE.L $70,vblankpoke+2        ; VBLANK and insert into new versions
 MOVE.W #0,count                ; Counter is set to zero
 LEA message,A2                 ; A2 = pos of text
 MOVE.L A2,D0                   ; A2 = start of free memory
 ADD.L #512,D0                  ; Force it to a 512 byte boundry
 AND.L #$FFFFFE00,D0
 MOVE.L D0,$044E(A5)            ; And that is the LOW screen
 MOVE.L D0,low                  ; Set LOW
 MOVE.L #whoops,$046E(A5)
 MOVE.W #$0000,-(SP)            ; Hardware and Software to low
 MOVE.L #-1,-(SP)
 MOVE.L #-1,-(SP)
 MOVE.W #5,-(SP)
 TRAP #14                       ; Set low resolution
whoops:
 ADD.L #12,SP
 MOVE.L $70,A0                  ; Save the old VBLANK
 MOVE.L #simple_vblank,$70      ; And set up mine
 MOVE.W #1,raster_flag          ; Set raster flag to 'not occured yet'
wait_raster:
 TST.W raster_flag              ; If still not occured
 BNE wait_raster                ; then loop until a vbl does occur
 MOVE.B #2,$FF8260              ; Back to HIGH hardware after VBL
 MOVE.L A0,$70
```

```
 MOVE.L #xbios,$B8              ; Set up the new XBIOS vector
 MOVE.L #vblank,$70             ; And the new VBLANK vector
 MOVE.L savestack,-(SP)         ; Restore the Supervisor stack
 MOVE.W #32,-(SP)               ; And go back to User mode
 TRAP #1
 ADDQ.L #6,SP
 MOVE.L (SP)+,D0                ; Tidy stack
 CLR.W -(SP)                    ; Exit ok for GEM
 MOVE.L D0,-(SP)                ; Length of program + data space
 MOVE.W #$31,-(SP)              ; terminate and stay resident (TSR)
 TRAP #1                        ; Finished this AUTO program

xbios:
 MOVEM.L A1/A2,-(SP)            ; Save A1 and A2
 MOVE.L SP,A2                   ; A2 = the stack
 ADD.L #8,A2                    ; offset over A1 and A2
 BTST #5,(A2)                   ; Test if called from user mode
 BNE notuser                    ; Skip if it is
 MOVE.L USP,A2                  ; Otherwise get A2 = User stack
 SUB.L #6,A2                    ; Offset it as if it were the SSP
notuser:
 MOVE.W $6(A2),D0               ; Get XBIOS instruction code
 CMP.W #2,D0                    ; If it is _PHYSBASE
 BEQ physbase                   ; then jump to new PHYSBASE routine
 CMP.W #4,D0                    ; If it is _GETREZ
 BEQ getrez                     ; then jump to new GETREZ routine
 CMP.W #5,D0                    ; If it is NOT _SETSCREEN
 BNE norm_xbios                 ; Then continue with the normal XBIOS
 MOVE.W #-1,16(A2)              ; Else alter rez.W to -1 (No change)
 MOVE.L 12(A2),D0               ; Get the ploc.L parameter
 CMP.L #-1,D0                   ; If it is -1
 BEQ norm_xbios                 ; then continue with normal XBIOS
 MOVE.L D0,low                  ; Otherwise, new value goes to LOW
 MOVE.L #-1,12(A2)              ; Set ploc.L to -1 (no change)
 BRA norm_xbios                 ; then norm BIOS deals with lloc.L

physbase:
 MOVE.L low,D0                  ; Get address of LOW screen
 MOVEM.L (SP)+,A1/A2            ; Tidy stack
 RTE                            ; Return LOW screen location

getrez:
 MOVE.W #0,D0                   ; Pretend we are in LOW resolution
 MOVEM.L (SP)+,A1/A2            ; Tidy the stack
 RTE                            ; Return code for LOW resolution

norm_xbios:
 MOVEM.L (SP)+,A1/A2            ; Tidy the stack up
xbiospoke:
 JMP $0.L                       ; And jump into the normal XBIOS

vblank:
 TST.W $43E                     ; Test flock system variable
```

```
 BNE vblankpoke                 ; No update if drive active

 MOVEM.L D0-D7/A0-A6,-(SP)      ; Save all registers
 CLR.L D0
 MOVE.B $FF8201,D0              ; Video base high
 LSL.L #8,D0                    ; times 256
 MOVE.B $FF8203,D0              ; Plus video base low
 LSL.L #8,D0                    ; All times 256
 MOVE.L D0,A3                   ; Is the address of the Real screen
 MOVE.L low,A2                  ; A0 = virtual LOW screen
 MOVE.L high,A1                 ; A1 = real HIGH screen
 CMP.L A1,A3                    ; Check if the real screen has moved
 BEQ screenok                   ; Skip this if not
 MOVE.L A3,A2                   ; Get the new real screen address
 MOVE.L A2,low                  ; Set LOW From this
 MOVE.L A1,D0                   ; And put the real screen back
 LSR.L #8,D0                    ; to its origional position
 MOVE.B D0,$FF8203
 LSR.L #8,D0
 MOVE.B D0,$FF8201

screenok:
 MOVE.L count,D2                ; How far did we do last time
 ADD.L D2,A1                    ; Move to this spot
 ADD.L D2,A2
 MOVE.B step,D3                 ; Get preset speed

mainloop:
 ADD.L #160,D2                  ; 160 bytes / color line
 CMP.L #32000,D2                ; 32000 bytes / screen
 BNE go_on
 CLR.W D2                       ; Start at the top once more
 MOVE.L high,A1
go_on:
 MOVE.B #20,D7                  ; 20 times / line
lineloop:
 ADDQ.L #4,A2
 MOVE.W (A2)+,D5
 MOVE.W (A2)+,D4
 MOVE.B #4,D0                   ; 4 times / color-byte
wordloop:
 ROXL.W #1,D4                   ; 4 color pixels
 ROXL.B #1,D6                   ; to one mono-byte
 ROXL.W #1,D5
 ROXL.B #1,D6
 ROXL.W #1,D4                   ; 2nd pixel
 ROXL.B #1,D6
 ROXL.W #1,D5
 ROXL.B #1,D6
 ROXL.W #1,D4                   ; 3rd pixel
 ROXL.B #1,D6
 ROXL.W #1,D5
 ROXL.B #1,D6
```

```
 ROXL.W #1,D4                    ; 4th pixel
 ROXL.B #1,D6
 ROXL.W #1,D5
 ROXL.B #1,D6

 MOVE.B D6,80(A1)                ; Set mono-byte  (low  line)
 MOVE.B D6,(A1)+                 ; Set mono-byte  (high line)
 SUBQ.B #1,D0
 BNE wordloop
 SUBQ.B #1,D7
 BNE lineloop
 ADD.L #80,A1
 SUBQ.B #1,D3
 BNE mainloop

 MOVE.L D2,count
 MOVEM.L (SP)+,D0-D7/A0-A6       ; Restore all registers
vblankpoke:
 JMP $0.L                        ; Jump to normal VBLANK routine

simple_vblank:
 CLR.W raster_flag               ; Indicate a Vertical blank has occured
 RTE

 ALIGN.W

savestack:   DC.L 0
low:         DC.L 0             ; Base address of LOW screen
high:        DC.L 0             ; Base address of HIGH screen
count:       DC.L 0             ; Count so far
raster_flag: DC.W 0             ; Flag cleared every raster (tempory)
step:        DC.B 0             ; Speed (counts / vblank)

message:
 DC.B 27,'E',10,'  The Color Emulator v 1.0',13,10,10
 DC.B '  Should be in AUTO Folder',13,10
 DC.B '  Will run with any TOS',13,10,10
 DC.B '  Author:  Lars-Erik 2sterud',13,10,10
 DC.B '  Adress:  2kriveien 39',13,10
 DC.B '           N-1349 Rykkinn',13,10
 DC.B '           NORWAY',13,10,10
 DC.B '  Phone:   +47 2 13 15 71',13,10
 DC.B '  BBS:     +47 2 13 26 59',13,10,10
 DC.B '  Email:   leoe',64,'ifi.uio.no',13,10
 DC.B '           f-leoe',64,'ifi.uio.no',13,10,10,0

input:
 DC.B 13,10
 DC.B '  Enter update-speed (1 to 6): ',0

 END
```

From -5020285466642289375
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,cdbc1d707afb026b,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-20 16:30:01 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!rutgers!apple!sun-barr!newstop!sun!
grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i056: gccgem -- GEM libraries for GCC part01/06
Keywords: shar, C
Message-ID: <33904@grapevine.uucp>
Date: 20 Jul 89 23:30:01 GMT
Sender: news@grapevine.uucp
Lines: 1410
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: uunet!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 2, Issue 56
Archive-name: gccgem/part01

This is the long awaited GEM library for GNU C. This code would also work
for other C-compilers. Watch out for the inline assembler statements in
gem-guts.h. You may need to replace them by whatever your compiler supports.

I am posting this for a friend who has no access to the net.
If there are any questions mail to me at roeder@sbsvax.uucp and i will
forward them to him.

[I repackaged this, as the original submission was in 60K-parts. -sg]

- Edgar

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
# This is part 1 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
#
#    Run the following text with /bin/sh to create:
#      COPYING
#      README
#      aes.h
#      aes_10.c
#      aes_100.c
#      aes_101.c
#      aes_102.c
#      aes_103.c
#      aes_104.c
#      aes_105.c
#      aes_106.c
#      aes_107.c
#      aes_108.c
```

```
#      aes_11.c
#      aes_110.c
#      aes_111.c
#      aes_112.c
#      aes_113.c
#      aes_114.c
#      aes_12.c
#      aes_120.c
#      aes_121.c
#      aes_122.c
#      aes_123.c
#      aes_124.c
#      aes_125.c
#      aes_13.c
#      aes_14.c
#      aes_15.c
#      aes_19.c
#      aes_20.c
#      aes_21.c
#      aes_22.c
#      aes_23.c
#      aes_24.c
#      aes_25.c
#      aes_26.c
#      aes_30.c
#      aes_31.c
#      aes_32.c
#      aes_33.c
#      aes_34.c
#      aes_35.c
#      aes_40.c
#      aes_41.c
#      aes_42.c
#      aes_43.c
#      aes_44.c
#      aes_45.c
#      aes_46.c
#      aes_47.c
#      aes_50.c
#      aes_51.c
#      aes_52.c
#      aes_53.c
#      aes_54.c
#      aes_55.c
#      aes_56.c
#      aes_70.c
#      aes_71.c
#      aes_72.c
#      aes_73.c
#      aes_74.c
#      aes_75.c
#      aes_76.c
#      aes_77.c
```

```
#       aes_78.c
#       aes_79.c
#       aes_80.c
#       aes_81.c
#       aes_90.c
#       aes_91.c
#       felder.c
#       gem-guts.h
#       vdi.h
#       vdi11_10.c
#       vdi5_1.c
#       vdi5_10.c
#       vdi5_100.c
#       vdi5_101.c
#       vdi5_102.c
#       vdi5_11.c
#       vdi5_12.c
#       vdi5_13.c
#       vdi5_14.c
#       vdi5_15.c
#       vdi5_16.c
#       vdi5_17.c
#       vdi5_18.c
#       vdi5_19.c
#       vdi5_2.c
#       vdi5_20.c
#       vdi5_21.c
#       vdi5_22.c
#       vdi5_23.c
#       vdi5_3.c
#       vdi5_4.c
#       vdi5_5.c
#       vdi5_6.c
#       vdi5_7.c
#       vdi5_8.c
#       vdi5_9.c
#       vdi5_98.c
#       vdi5_99.c
#       vdi_1.c
#       vdi_10.c
#       vdi_100.c
#       vdi_101.c
#       vdi_102.c
#       vdi_103.c
#       vdi_104.c
#       vdi_105.c
#       vdi_106.c
#       vdi_107.c
#       vdi_108.c
#       vdi_109.c
#       vdi_110.c
#       vdi_111.c
#       vdi_112.c
```

```
#      vdi_113.c
#      vdi_114.c
#      vdi_115.c
#      vdi_116.c
#      vdi_117.c
#      vdi_118.c
#      vdi_119.c
#      vdi_11_1.c
#      vdi_11_2.c
#      vdi_11_3.c
#      vdi_11_4.c
#      vdi_11_5.c
#      vdi_11_6.c
#      vdi_11_7.c
#      vdi_11_8.c
#      vdi_11_9.c
#      vdi_12.c
#      vdi_120.c
#      vdi_121.c
#      vdi_122.c
#      vdi_123.c
#      vdi_124.c
#      vdi_125.c
#      vdi_126.c
#      vdi_127.c
#      vdi_128.c
#      vdi_129.c
#      vdi_13.c
#      vdi_130.c
#      vdi_131.c
#      vdi_14.c
#      vdi_15.c
#      vdi_16.c
#      vdi_17.c
#      vdi_18.c
#      vdi_19.c
#      vdi_2.c
#      vdi_20.c
#      vdi_21.c
#      vdi_22.c
#      vdi_23.c
#      vdi_24.c
#      vdi_25.c
#      vdi_26.c
#      vdi_27.c
#      vdi_28.c
#      vdi_29.c
#      vdi_3.c
#      vdi_30.c
#      vdi_31.c
#      vdi_32.c
#      vdi_33.c
#      vdi_35.c
```

```
#      vdi_36.c
#      vdi_37.c
#      vdi_38.c
#      vdi_39.c
#      vdi_4.c
#      vdi_6.c
#      vdi_7.c
#      vdi_8.c
#      vdi_9.c
#      vdi_b.c
#      vdi_c.c
#      vdi_d.c
#      vdi_e.c
#      vdi_m_a.c
#
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
echo "x - extracting COPYING (Text)"
sed 's/^X//' << 'SHAR_EOF' > COPYING &&
X
X            GNU GENERAL PUBLIC LICENSE
X             Version 1, February 1989
X
X Copyright (C) 1989 Free Software Foundation, Inc.
X                    675 Mass Ave, Cambridge, MA 02139, USA
X Everyone is permitted to copy and distribute verbatim copies
X of this license document, but changing it is not allowed.
X
X                Preamble
X
X  The license agreements of most software companies try to keep users
Xat the mercy of those companies.  By contrast, our General Public
XLicense is intended to guarantee your freedom to share and change free
Xsoftware--to make sure the software is free for all its users.  The
XGeneral Public License applies to the Free Software Foundation's
Xsoftware and to any other program whose authors commit to using it.
XYou can use it for your programs, too.
X
X  When we speak of free software, we are referring to freedom, not
Xprice.  Specifically, the General Public License is designed to make
Xsure that you have the freedom to give away or sell copies of free
Xsoftware, that you receive source code or can get it if you want it,
Xthat you can change the software or use pieces of it in new free
Xprograms; and that you know you can do these things.
X
X  To protect your rights, we need to make restrictions that forbid
Xanyone to deny you these rights or to ask you to surrender the rights.
XThese restrictions translate to certain responsibilities for you if you
Xdistribute copies of the software, or if you modify it.
X
X  For example, if you distribute copies of a such a program, whether
```

Xgratis or for a fee, you must give the recipients all the rights that
Xyou have. You must make sure that they, too, receive or can get the
Xsource code. And you must tell them their rights.
X
X We protect your rights with two steps: (1) copyright the software, and
X(2) offer you this license which gives you legal permission to copy,
Xdistribute and/or modify the software.
X
X Also, for each author's protection and ours, we want to make certain
Xthat everyone understands that there is no warranty for this free
Xsoftware. If the software is modified by someone else and passed on, we
Xwant its recipients to know that what they have is not the original, so
Xthat any problems introduced by others will not reflect on the original
Xauthors' reputations.
X
X The precise terms and conditions for copying, distribution and
Xmodification follow.
X

X   GNU GENERAL PUBLIC LICENSE
X   TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION
X
X 0. This License Agreement applies to any program or other work which
Xcontains a notice placed by the copyright holder saying it may be
Xdistributed under the terms of this General Public License. The
X"Program", below, refers to any such program or work, and a "work based
Xon the Program" means either the Program or any work containing the
XProgram or a portion of it, either verbatim or with modifications. Each
Xlicensee is addressed as "you".
X
X 1. You may copy and distribute verbatim copies of the Program's source
Xcode as you receive it, in any medium, provided that you conspicuously and
Xappropriately publish on each copy an appropriate copyright notice and
Xdisclaimer of warranty; keep intact all the notices that refer to this
XGeneral Public License and to the absence of any warranty; and give any
Xother recipients of the Program a copy of this General Public License
Xalong with the Program. You may charge a fee for the physical act of
Xtransferring a copy.
X
X 2. You may modify your copy or copies of the Program or any portion of
Xit, and copy and distribute such modifications under the terms of Paragraph
X1 above, provided that you also do the following:
X
X   a) cause the modified files to carry prominent notices stating that
X   you changed the files and the date of any change; and
X
X   b) cause the whole of any work that you distribute or publish, that
X   in whole or in part contains the Program or any part thereof, either
X   with or without modifications, to be licensed at no charge to all
X   third parties under the terms of this General Public License (except
X   that you may choose to grant warranty protection to some or all
X   third parties, at your option).
X

X   c) If the modified program normally reads commands interactively when
X   run, you must cause it, when started running for such interactive use
X   in the simplest and most usual way, to print or display an
X   announcement including an appropriate copyright notice and a notice
X   that there is no warranty (or else, saying that you provide a
X   warranty) and that users may redistribute the program under these
X   conditions, and telling the user how to view a copy of this General
X   Public License.
X
X   d) You may charge a fee for the physical act of transferring a
X   copy, and you may at your option offer warranty protection in
X   exchange for a fee.
X
XMere aggregation of another independent work with the Program (or its
Xderivative) on a volume of a storage or distribution medium does not bring
Xthe other work under the scope of these terms.
X

X 3. You may copy and distribute the Program (or a portion or derivative of
Xit, under Paragraph 2) in object code or executable form under the terms of
XParagraphs 1 and 2 above provided that you also do one of the following:
X
X   a) accompany it with the complete corresponding machine-readable
X   source code, which must be distributed under the terms of
X   Paragraphs 1 and 2 above; or,
X
X   b) accompany it with a written offer, valid for at least three
X   years, to give any third party free (except for a nominal charge
X   for the cost of distribution) a complete machine-readable copy of the
X   corresponding source code, to be distributed under the terms of
X   Paragraphs 1 and 2 above; or,
X
X   c) accompany it with the information you received as to where the
X   corresponding source code may be obtained. (This alternative is
X   allowed only for noncommercial distribution and only if you
X   received the program in object code or executable form alone.)
X
XSource code for a work means the preferred form of the work for making
Xmodifications to it. For an executable file, complete source code means
Xall the source code for all modules it contains; but, as a special
Xexception, it need not include source code for modules which are standard
Xlibraries that accompany the operating system on which the executable
Xfile runs, or for standard header files or definitions files that
Xaccompany that operating system.
X
X 4. You may not copy, modify, sublicense, distribute or transfer the
XProgram except as expressly provided under this General Public License.
XAny attempt otherwise to copy, modify, sublicense, distribute or transfer
Xthe Program is void, and will automatically terminate your rights to use
Xthe Program under this License. However, parties who have received
Xcopies, or rights to use copies, from you under this General Public
XLicense will not have their licenses terminated so long as such parties
Xremain in full compliance.

X
X 5. By copying, distributing or modifying the Program (or any work based
Xon the Program) you indicate your acceptance of this license to do so,
Xand all its terms and conditions.
X
X 6. Each time you redistribute the Program (or any work based on the
XProgram), the recipient automatically receives a license from the original
Xlicensor to copy, distribute or modify the Program subject to these
Xterms and conditions. You may not impose any further restrictions on the
Xrecipients' exercise of the rights granted herein.
X

X 7. The Free Software Foundation may publish revised and/or new versions
Xof the General Public License from time to time. Such new versions will
Xbe similar in spirit to the present version, but may differ in detail to
Xaddress new problems or concerns.
X
XEach version is given a distinguishing version number. If the Program
Xspecifies a version number of the license which applies to it and "any
Xlater version", you have the option of following the terms and conditions
Xeither of that version or of any later version published by the Free
XSoftware Foundation. If the Program does not specify a version number of
Xthe license, you may choose any version ever published by the Free Software
XFoundation.
X
X 8. If you wish to incorporate parts of the Program into other free
Xprograms whose distribution conditions are different, write to the author
Xto ask for permission. For software which is copyrighted by the Free
XSoftware Foundation, write to the Free Software Foundation; we sometimes
Xmake exceptions for this. Our decision will be guided by the two goals
Xof preserving the free status of all derivatives of our free software and
Xof promoting the sharing and reuse of software generally.
X
X   NO WARRANTY
X
X 9. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY
XFOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN
XOTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES
XPROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED
XOR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
XMERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS
XTO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE
XPROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING,
XREPAIR OR CORRECTION.
X
X 10. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING
XWILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR
XREDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES,
XINCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING
XOUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED
XTO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY
XYOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER
XPROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE

XPOSSIBILITY OF SUCH DAMAGES.
X
X   END OF TERMS AND CONDITIONS
X

X   Appendix: How to Apply These Terms to Your New Programs
X
X If you develop a new program, and you want it to be of the greatest
Xpossible use to humanity, the best way to achieve this is to make it
Xfree software which everyone can redistribute and change under these
Xterms.
X
X To do so, attach the following notices to the program. It is safest to
Xattach them to the start of each source file to most effectively convey
Xthe exclusion of warranty; and each file should have at least the
X"copyright" line and a pointer to where the full notice is found.
X
X   <one line to give the program's name and a brief idea of what it does.>
X   Copyright (C) 19yy <name of author>
X
X   This program is free software; you can redistribute it and/or modify
X   it under the terms of the GNU General Public License as published by
X   the Free Software Foundation; either version 1, or (at your option)
X   any later version.
X
X   This program is distributed in the hope that it will be useful,
X   but WITHOUT ANY WARRANTY; without even the implied warranty of
X   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
X   GNU General Public License for more details.
X
X   You should have received a copy of the GNU General Public License
X   along with this program; if not, write to the Free Software
X   Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X
XAlso add information on how to contact you by electronic and paper mail.
X
XIf the program is interactive, make it output a short notice like this
Xwhen it starts in an interactive mode:
X
X   Gnomovision version 69, Copyright (C) 19xx name of author
X   Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
X   This is free software, and you are welcome to redistribute it
X   under certain conditions; type `show c' for details.
X
XThe hypothetical commands `show w' and `show c' should show the
Xappropriate parts of the General Public License. Of course, the
Xcommands you use may be called something other than `show w' and `show
Xc'; they could even be mouse-clicks or menu items--whatever suits your
Xprogram.
X
XYou should also get your employer (if you work as a programmer) or your
Xschool, if any, to sign a "copyright disclaimer" for the program, if
Xnecessary. Here a sample; alter the names:

```
X
X  Yoyodyne, Inc., hereby disclaims all copyright interest in the
X  program `Gnomovision' (a program to direct compilers to make passes
X  at assemblers) written by James Hacker.
X
X  <signature of Ty Coon>, 1 April 1989
X  Ty Coon, President of Vice
X
XThat's all there is to it!
SHAR_EOF
chmod 0644 COPYING || echo "restore of COPYING fails"
echo "x - extracting README (Text)"
sed 's/^X//' << 'SHAR_EOF' > README &&
XI am posting this for a friend who has no access to the net. He has hacked in
Xmost of the VDI and AES functions used on the ST. If there are any questions
Xmail to me at roeder@sbsvax.uucp and i will forward them to him.
X
XThis library may be distributed under the terms of the GNU General Public
XLicence which can be found in the file COPYING.
SHAR_EOF
chmod 0644 README || echo "restore of README fails"
echo "x - extracting aes.h (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes.h &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X/*
X *      AES.H - Headerfile fuer die AES Bindings
X *
X *      version 1.0       13.7.89  stefan brueck (sb)
X */
X
X#ifndef __AES_H
X#define __AES_H
X
X#include <portab.h>
```

```
X
Xextern WORD global[];
Xextern WORD control[];
Xextern WORD int_in[];
Xextern WORD int_out[];
Xextern LONG addr_in[];
Xextern LONG addr_out[];
X
Xtypedef struct {
X    WORD *cb_pcontrol;
X    WORD *cb_pglobal;
X    WORD *cb_pintin;
X    WORD *cb_pintout;
X    LONG *cb_padrin;
X    LONG *cb_padrout;
X    } AESPB;
X
Xextern WORD gl_apid;
Xextern AESPB aespb;
X
Xtypedef struct {
X         WORD ob_next;
X         WORD ob_head;
X         WORD ob_tail;
X         unsigned WORD      ob_type;
X         unsigned WORD      ob_flags;
X         unsigned WORD      ob_state;
X         char *ob_spec;
X         WORD ob_x;
X         WORD ob_y;
X         WORD ob_width;
X         WORD ob_height;
X    } OBJECT;
X
Xtypedef struct {
X         char *te_ptext;
X         char *te_ptmplt;
X         char *te_pvalid;
X         WORD te_font;
X         WORD te_resvd1;
X         WORD te_just;
X         WORD te_color;
X         WORD te_resvd2;
X         WORD te_thickness;
X         WORD te_txtlen;
X         WORD te_tmplen;
X    } TEDINFO;
X
Xtypedef struct {
X         WORD *ib_pmask;
X         WORD *ib_pdata;
X         char *ib_ptext;
X         WORD ib_char;
```

```
X         WORD ib_xchar;
X         WORD ib_ychar;
X         WORD ib_xicon;
X         WORD ib_yicon;
X         WORD ib_wicon;
X         WORD ib_hicon;
X         WORD ib_xtext;
X         WORD ib_ytext;
X         WORD ib_wtext;
X         WORD ib_htext;
X         WORD ib_resvd;
X    } ICONBLK;
X
Xtypedef struct {
X         WORD *bi_pdata;
X         WORD bi_wb;
X         WORD bi_hl;
X         WORD bi_x;
X         WORD bi_y;
X         WORD bi_color;
X    } BITBLK;
X
Xtypedef struct {
X         WORD (*ub_code)();
X         LONG ub_parm;
X    } USERBLK;
X
Xtypedef struct {
X         OBJECT   *pb_tree;
X         WORD pb_obj;
X         WORD pb_prevstate;
X         WORD pb_currstate;
X         WORD pb_x;
X         WORD pb_y;
X         WORD pb_w;
X         WORD pb_h;
X         WORD pb_xc;
X         WORD pb_yc;
X         WORD pb_wc;
X         WORD pb_hc;
X         LONG pb_parm;
X    } PARMBLK;
X
Xtypedef struct {
X         WORD mf_xhot;
X         WORD mf_yhot;
X         WORD mf_nplanes;
X         WORD mf_fg;
X         WORD mf_bg;
X         WORD mf_mask[16];
X         WORD mf_data[16];
X    } MFORM;
X
```

```
X/***************** application-library ********************/
X
XWORD    appl_init();
XWORD    appl_read( WORD ap_rid, WORD ap_rlength, char *ap_rbuff);
XWORD    appl_write( WORD ap_wid, WORD ap_wlength, char *ap_wbuff);
XWORD    appl_find( char *ap_fname);
XWORD    appl_tplay( char *ap_tpmem, WORD ap_tpnum, WORD ap_tpscale);
XWORD    appl_trecord( char *ap_trmem, WORD ap_trcount);
XWORD    appl_exit();
X
X/***************** event-library ************************/
X
XWORD    evnt_keybd();
XWORD    evnt_button( WORD ev_bclicks, WORD ev_bmask, WORD ev_bstate, WORD
*ev_bmx, WORD *ev_bmy, WORD *ev_bbutton, WORD *ev_bkstate);
XWORD    evnt_mouse( WORD ev_moflags, WORD ev_mox, WORD ev_moy, WORD ev_mowidth,
WORD ev_moheight, WORD *ev_momx, WORD *ev_momy, WORD *ev_mobutton, WORD
*ev_mokstate);
XWORD    evnt_mesag(char *ev_mgpbuff);
XWORD    evnt_timer(unsigned WORD ev_tlocount, unsigned WORD ev_thicount);
XWORD    evnt_multi( WORD a, WORD b, WORD c, WORD d, WORD e, WORD f, WORD g, WORD
h, WORD i, WORD j, WORD k, WORD l, WORD m, WORD n, char *buff, WORD o, WORD p,
WORD *q, WORD *r, WORD *s, WORD *t, WORD *u, WORD *v );
XWORD    evnt_dclick( WORD ev_dnew, WORD ev_dgetset);
X
X/***************** menu-library ***************************/
X
XWORD    menu_bar( OBJECT * me_btree, WORD me_bshow);
XWORD    menu_icheck( OBJECT * me_ctree, WORD me_citem, WORD me_ccheck);
XWORD    menu_ienable( OBJECT * me_etree, WORD me_eitem, WORD me_eenable);
XWORD    menu_tnormal( OBJECT * me_ntree, WORD me_ntitle, WORD me_nnormal);
XWORD    menu_text( OBJECT * me_ttree, WORD me_titem, char *me_ttext);
XWORD    menu_register( WORD me_rapid, char *me_rpstring);
X
X/***************** object-library ************************/
X
XWORD    objc_add( OBJECT * ab_atree, WORD ob_parent, WORD ob_achild);
XWORD    objc_delete( OBJECT * ob_dltree, WORD ob_dlobject);
XWORD    objc_draw( OBJECT * ob_drtree, WORD ob_drstartob, WORD ob_drdepth, WORD
ob_drxclip, WORD ob_dryclip, WORD ob_drwclip, WORD ob_drhclip);
XWORD    objc_find( OBJECT * ob_ftree, WORD ob_fstartob, WORD ob_fdepth, WORD
ob_fmx, WORD ob_fmy);
XWORD    objc_offset( OBJECT * ob_oftree, WORD ob_ofobject, WORD *ob_ofxoff, WORD
*ob_ofyoff);
XWORD    objc_order( OBJECT * ob_ortree, WORD ob_orobject, WORD ob_ornewpos);
XWORD    objc_edit( OBJECT * ob_edtree, WORD ob_edobject, WORD ob_edchar, WORD
*ob_edidx, WORD *ob_edkind);
XWORD    objc_change( OBJECT * ob_ctree, WORD ob_cobject, WORD ob_cresvd, WORD
ob_cxclip, WORD ob_cyclip, WORD ob_cwclip, WORD ob_chclip, WORD ob_cnewstate, WORD
ob_credraw);
X
X/********************* form-library **********************/
X
```

```
XWORD    form_do( OBJECT * fo_dotree, WORD fo_dostartob);
XWORD    form_dial( WORD fo_diflag, WORD fo_dilittlx, WORD fo_dilittly, WORD
fo_dilittlw, WORD fo_dilittlh, WORD fo_dibigx, WORD fo_dibigy, WORD fo_dibigw,
WORD fo_dibigh);
XWORD    form_alert( WORD fo_adefbttn, char *fo_astring);
XWORD    form_error( WORD fo_enum);
XWORD    form_center( OBJECT * fo_ctree, WORD *fo_cx, WORD *fo_cy, WORD *fo_cw,
WORD *fo_ch);
XWORD    form_keybd( OBJECT * fo_ktree, WORD fo_kobject, WORD fo_kobnext, WORD
fo_kchar, WORD *fo_knxtobject, WORD *fo_nxtchar);
XWORD    formbutton( OBJECT * fo_btree, WORD fo_bobject, WORD fo_bclicks, WORD
*fo_bnxtobj);
X
X/******************** graf-library ***********************/
X
XWORD    graf_rubberbox( WORD gr_rx, WORD gr_ry, WORD gr_rw, WORD gr_rh, WORD
*gr_rlastwidth, WORD *gr_rlastheight);
XWORD    graf_dragbox( WORD gr_dwidth, WORD gr_dheight, WORD gr_dstartx, WORD
gr_dstarty, WORD gr_dboundx, WORD gr_dboundy, WORD gr_dboundw, WORD gr_dboundh,
WORD *gr_dfinishx, WORD *gr_dfinishy);
XWORD    graf_movebox( WORD gr_mwidth, WORD gr_mheight, WORD gr_msourcex, WORD
gr_msourcey, WORD gr_mdestx, WORD gr_mdesty);
XWORD    graf_growbox( WORD gr_gstx, WORD gr_gsty, WORD gr_gstwidth, WORD
gr_gstheight, WORD gr_gfinx, WORD gr_gfiny, WORD gr_gfinwidth, WORD
gr_gfinheight);
XWORD    graf_shrinkbox( WORD gr_sfinx, WORD gr_sfiny, WORD gr_sfinwidth, WORD
gr_sfinheight, WORD gr_sstx, WORD gr_ssty, WORD gr_sstwidth, WORD gr_sstheight);
XWORD    graf_watchbox( OBJECT * gr_wptree, WORD gr_wobject, WORD gr_winstate,
WORD gr_woutstate);
XWORD    graf_slidebox( OBJECT * gr_sltree, WORD gr_slobject, WORD gr_slvh);
XWORD    graf_handle( WORD *gr_hwchar, WORD *gr_hhchar, WORD *gr_hwbox, WORD
*gr_hhbox);
XWORD    graf_mouse( WORD gr_monumber, MFORM *gr_mofaddr);
XWORD    graf_mkstate( WORD *gr_mkmx, WORD *gr_mkmy, WORD *gr_mkmstate, WORD
*gr_mkkstate);
X
X/******************* scrap-library ************************/
X
XWORD    scrp_read( char *sc_rpscrap);
XWORD    scrp_write( char *sc_wpscrap);
X
X/******************* fsel-library *************************/
X
XWORD    fsel_input( char *fs_iinpath, char* fs_iinsel, WORD *fs_iexbutton);
XWORD    fsel_exinput(char *fs_iinpath, char *fs_iinsel, WORD *fs_iexbutton, char
*fs_label);
X
X/******************* window-library ***********************/
X
XWORD    wind_create( WORD wi_crkind, WORD wi_crwx, WORD wi_crwy, WORD wi_crww,
WORD wi_crwh);
XWORD    wind_open( WORD wi_ohandle, WORD wi_owx, WORD wi_owy, WORD wi_oww, WORD
wi_owh);
```

```
XWORD    wind_close( WORD wi_clhandle);
XWORD    wind_delete( WORD wi_dhandle);
XWORD    wind_get( WORD wi_ghandle, WORD wi_gfield, WORD *wi_gw1, WORD *wi_gw2,
WORD *wi_gw3, WORD *wi_gw4);
XWORD    wind_set( WORD wi_shandle, WORD wi_sfield, WORD wi_sw1, WORD wi_sw2, WORD
wi_sw3, WORD wi_sw4);
XWORD    wind_find( WORD wi_fmx, WORD wi_fmy);
XWORD    wind_update( WORD wi_ubegend);
XWORD    wind_calc( WORD wi_ctype, WORD wi_ckind, WORD wi_cinx, WORD wi_ciny, WORD
wi_cinw, WORD wi_cinh, WORD *wi_coutx, WORD *wi_couty, WORD *wi_coutw, WORD
*wi_couth);
X
X/****************** resource-library ************************/
X
XWORD    rsrc_load(char *re_lpfname);
XWORD    rsrc_free();
XWORD    rsrc_gaddr( WORD re_gtype, WORD re_gindex, OBJECT **re_gaddr);
XWORD    rsrc_saddr( WORD re_stype, WORD re_sindex, LONG re_saddr);
XWORD    rsrc_obfix( OBJECT * re_otree, WORD re_oobject);
X
X/****************** shel-library ****************************/
X
XWORD    shel_read( char *sh_rpread, char *sh_rptail);
XWORD    shel_write( WORD sh_wdoex, WORD sh_wisgr, WORD sh_wiscr, char *sh_wpcmd,
char *sh_wptail);
XWORD    shel_get( char *sh_gaddr, WORD sh_glen);
XWORD    shel_put( char *sh_paddr, WORD sh_plen);
XWORD    shel_find( char *sh_fpbuff);
XWORD    shel_envrn( char *sh_epvalue, char *eparm);
X
X/************************************************************/
X
X#define MN_SELECTED        10
X#define WM_REDRAW          20
X#define WM_TOPPED          21
X#define WM_CLOSED          22
X#define WM_FULLED          23
X#define WM_ARROWED         24
X#define WM_HSLID           25
X#define WM_VSLID           26
X#define WM_SIZED           27
X#define WM_MOVED           28
X#define WM_NEWTOP          29
X
X#define AC_OPEN            40
X#define AC_CLOSE           41
X
X#define LEFT_MOUSE_BUTTON  0x0001
X#define RIGHT_MOUSE_BUTTON 0x0002
X#define RIGHT_SHIFT_KEY         0x0001
X#define LEFT_SHIFT_KEY     0x0002
X#define CONTROL_KEY             0x0004
X#define ALTERNATE_KEY      0x0008
```

```
X
X/* fensterelemente */
X
X#define NAME                    0x0001
X#define CLOSER              0x0002
X#define FULLER              0x0004
X#define MOVER                   0x0008
X#define INFO                0x0010
X#define SIZER                   0x0020
X#define UPARROW             0x0040
X#define DOWNARROW           0x0080
X#define VSLID                   0x0100
X#define LEFTARROW           0x0200
X#define RIGHTARROW          0x0400
X#define HSLID                   0x0800
X
X/* fenstermanipulation */
X
X#define WC_BORDER           0
X#define WC_WORK             1
X
X#define WF_KIND             1
X#define WF_NAME             2
X#define WF_INFO             3
X#define WF_WORKXYWH         4
X#define WF_CURRXYWH         5
X#define WF_PREVXYWH         6
X#define WF_FULLXYWH         7
X#define WF_HSLID            8
X#define WF_VSLID            9
X#define WF_TOP              10
X#define WF_FIRSTXYWH   11
X#define WF_NEXTXYWH    12
X#define WF_RESVD            13
X#define WF_NEWDESK          14
X#define WF_HSLSIZE          15
X#define WF_VSLSIZE          16
X#define WF_SCREEN           17
X
X#define END_UPDATE          0
X#define BEG_UPDATE          1
X#define END_MCTRL           2
X#define BEG_MCTRL           3
X
X/* events */
X
X#define MU_KEYBD            0x0001
X#define MU_BUTTON           0x0002
X#define MU_M1                   0x0004
X#define MU_M2                   0x0008
X#define MU_MESAG            0x0010
X#define MU_TIMER            0x0020
X
```

```
X/* ob_type */
X
X#define G_BOX                   20
X#define G_TEXT                  21
X#define G_BOXTEXT           22
X#define G_IMAGE             23
X#define G_USERDEF           24
X#define G_IBOX                  25
X#define G_BUTTON            26
X#define G_BOXCHAR           27
X#define G_STRING            28
X#define G_FTEXT             29
X#define G_FBOXTEXT          30
X#define G_ICON                  31
X#define G_TITLE             32
X
X/* ob_flags */
X
X#define SELECTABLE      0x0001
X#define DEFAULT         0x0002
X#define EXIT            0x0004
X#define EDITABLE        0x0008
X#define RBUTTON         0x0010
X#define LASTOB              0x0020
X#define TOUCHEXIT       0x0040
X#define HIDETREE        0x0080
X#define INDIRECT        0x0100
X
X/* ob_state */
X
X#define SELECTED        0x0001
X#define CROSSED         0x0002
X#define CHECKED         0x0004
X#define DISABLED        0x0008
X#define OUTLINED        0x0010
X#define SHADOWED        0x0020
X
X/* objc_edit */
X
X#define ED_START        0
X#define ED_INIT         1
X#define ED_CHAR         2
X#define ED_END              3
X
X/* form_alert */
X
X#define NO_ICON         0
X#define NOTE            1
X#define WAIT            2
X#define STOP            3
X
X/* form_dial */
X
```

```
X#define FMD_START     0
X#define FMD_GROW      1
X#define FMD_SHRINK    2
X#define FMD_FINISH    3
X
X/* graf_mouse */
X
X#define ARROW              0
X#define TXT_CRSR      1
X#define HOURGLASS     2
X#define BUSYBEE       2
X#define POINT_HAND    3
X#define PLAT_HAND     4
X#define THIN_CROSS    5
X#define THICK_CROSS   6
X#define OUTL_CROSS    7
X#define USER_DEF      255
X#define M_OFF              256
X#define M_ON          257
X
X/* wind_calc */
X
X#define NAME      0x0001
X#define CLOSE          0x0002
X#define FULL      0x0004
X#define MOVE      0x0008
X#define INFO      0x0010
X#define SIZE      0x0020
X#define UPARROW   0x0040
X#define DOWNARROW 0x0080
X#define VSLIDE         0x0100
X#define LFARROW   0x0200
X#define RTARROW   0x0400
X#define HSLIDE         0x0800
X
X#endif /* not __AES_H */
X
SHAR_EOF
chmod 0644 aes.h || echo "restore of aes.h fails"
echo "x - extracting aes_10.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_10.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
```

```
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD gl_apid;
XAESPB aespb;
X
XWORD appl_init()
X{
X    aespb.cb_pcontrol = control;
X    aespb.cb_pglobal  = global;
X    aespb.cb_pintin   = int_in;
X    aespb.cb_pintout  = int_out;
X    aespb.cb_padrin   = addr_in;
X    aespb.cb_padrout  = addr_out;
X
X    CONTROL(10,0,1,0,0);
X    AES_CALL();
X    return (gl_apid = int_out[0]);
X}
SHAR_EOF
chmod 0644 aes_10.c || echo "restore of aes_10.c fails"
echo "x - extracting aes_100.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_100.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD wind_create(WORD wi_crkind, WORD wi_crwx, WORD wi_crwy, WORD wi_crww, WORD
```

```
wi_crwh)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = wi_crkind;
X    *a_tmp++ = wi_crwx;
X    *a_tmp++ = wi_crwy;
X    *a_tmp++ = wi_crww;
X    *a_tmp   = wi_crwh;
X    CONTROL(100,5,1,0,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_100.c || echo "restore of aes_100.c fails"
echo "x - extracting aes_101.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_101.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD wind_open(WORD wi_ohandle, WORD wi_owx, WORD wi_owy, WORD wi_oww, WORD
wi_owh)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = wi_ohandle;
X    *a_tmp++ = wi_owx;
X    *a_tmp++ = wi_owy;
X    *a_tmp++ = wi_oww;
X    *a_tmp   = wi_owh;
X    CONTROL(101,5,1,0,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_101.c || echo "restore of aes_101.c fails"
```

```
echo "x - extracting aes_102.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_102.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD wind_close(WORD wi_clhandle)
X{
X    int_in[0] = wi_clhandle;
X    CONTROL(102,1,1,0,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_102.c || echo "restore of aes_102.c fails"
echo "x - extracting aes_103.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_103.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
```

```
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD wind_delete(WORD wi_dhandle)
X{
X    int_in[0] = wi_dhandle;
X    CONTROL(103,1,1,0,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_103.c || echo "restore of aes_103.c fails"
echo "x - extracting aes_104.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_104.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD wind_get(WORD wi_ghandle, WORD wi_gfield, WORD *wi_gw1, WORD *wi_gw2, WORD
*wi_gw3, WORD *wi_gw4)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = wi_ghandle;
X    *a_tmp   = wi_gfield;
X    CONTROL(104,2,5,0,0);
X    AES_CALL();
X    a_tmp = int_out;
X    *wi_gw1 = *++a_tmp;
X    *wi_gw2 = *++a_tmp;
X    *wi_gw3 = *++a_tmp;
X    *wi_gw4 = *++a_tmp;
X    return int_out[0];
X}
SHAR_EOF
```

```
chmod 0644 aes_104.c || echo "restore of aes_104.c fails"
echo "x - extracting aes_105.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_105.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD wind_set(WORD wi_shandle, WORD wi_sfield, WORD wi_sw1, WORD wi_sw2, WORD
wi_sw3, WORD wi_sw4)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = wi_shandle;
X    *a_tmp++ = wi_sfield;
X    *a_tmp++ = wi_sw1;
X    *a_tmp++ = wi_sw2;
X    *a_tmp++ = wi_sw3;
X    *a_tmp   = wi_sw4;
X    CONTROL(105,6,1,0,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_105.c || echo "restore of aes_105.c fails"
echo "x - extracting aes_106.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_106.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
```

```
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD wind_find(WORD wi_fmx, WORD wi_fmy)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = wi_fmx;
X    *a_tmp   = wi_fmy;
X    CONTROL(106,2,1,0,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_106.c || echo "restore of aes_106.c fails"
echo "x - extracting aes_107.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_107.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD wind_update(WORD wi_ubegend)
X{
X    int_in[0] = wi_ubegend;
X    CONTROL(107,1,1,0,0);
X    AES_CALL();
X    return int_out[0];
```

```
X}
SHAR_EOF
chmod 0644 aes_107.c || echo "restore of aes_107.c fails"
echo "x - extracting aes_108.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_108.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD wind_calc(WORD wi_ctype, WORD wi_ckind, WORD wi_cinx, WORD wi_ciny, WORD
wi_cinw, WORD wi_cinh, WORD *wi_coutx, WORD *wi_couty, WORD *wi_coutw, WORD
*wi_couth)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = wi_ctype;
X    *a_tmp++ = wi_ckind;
X    *a_tmp++ = wi_cinx;
X    *a_tmp++ = wi_ciny;
X    *a_tmp++ = wi_cinw;
X    *a_tmp   = wi_cinh;
X    CONTROL(108,6,5,0,0);
X    AES_CALL();
X    a_tmp = int_out;
X    *wi_coutx = *++a_tmp;
X    *wi_couty = *++a_tmp;
X    *wi_coutw = *++a_tmp;
X    *wi_couth = *++a_tmp;
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_108.c || echo "restore of aes_108.c fails"
echo "x - extracting aes_11.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_11.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
```

```
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD appl_read(WORD ap_rid, WORD ap_rlength, char *ap_rbuff)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = ap_rid;
X    *a_tmp   = ap_rlength;
X    addr_in[0]= (LONG) ap_rbuff;
X    CONTROL(11,2,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_11.c || echo "restore of aes_11.c fails"
echo "x - extracting aes_110.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_110.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
```

```
X */
X#include "gem-guts.h"
X
XWORD rsrc_load(char *re_lpfname)
X{
X    addr_in[0] = (LONG) re_lpfname;
X    CONTROL(110,0,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_110.c || echo "restore of aes_110.c fails"
echo "x - extracting aes_111.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_111.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD rsrc_free()
X{
X    CONTROL(111,0,1,0,0);
X    AES_CALL();
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_111.c || echo "restore of aes_111.c fails"
echo "x - extracting aes_112.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_112.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
```

```
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD rsrc_gaddr(WORD re_gtype, WORD re_gindex, OBJECT **re_gaddr)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = re_gtype;
X    *a_tmp   = re_gindex;
X    CONTROL(112,2,1,0,1);
X    control[4] = 1;
X    AES_CALL();
X    control[4] = 0;
X    *re_gaddr = (OBJECT *) addr_out[0];
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_112.c || echo "restore of aes_112.c fails"
echo "x - extracting aes_113.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_113.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
SHAR_EOF
echo "End of part 1"
echo "File aes_113.c is continued in part 2"
echo "2" > s2_seq_.tmp
exit 0


From -751896382637334981
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,1665f42ab9099103,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-20 16:30:11 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!ames!apple!sun-barr!newstop!sun!
grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i057: gccgem -- GEM libraries for GCC  part02/06
Keywords: shar, C
```

```
Message-ID: <33905@grapevine.uucp>
Date: 20 Jul 89 23:30:11 GMT
Sender: news@grapevine.uucp
Lines: 1322
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: uunet!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 2, Issue 57
Archive-name: gccgem/part02

#!/bin/sh
# this is part 2 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file aes_113.c continued
#
CurArch=2
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file aes_113.c"
sed 's/^X//' << 'SHAR_EOF' >> aes_113.c
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD rsrc_saddr(WORD re_stype, WORD re_sindex, LONG re_saddr)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = re_stype;
X    *a_tmp   = re_sindex;
X    addr_in[0] = re_saddr;
```

```
X    CONTROL(113,2,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
echo "File aes_113.c is complete"
chmod 0644 aes_113.c || echo "restore of aes_113.c fails"
echo "x - extracting aes_114.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_114.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD rsrc_obfix(OBJECT *re_otree, WORD re_oobject)
X{
X    addr_in[0] = (LONG) re_otree;
X    int_in[0]  = re_oobject;
X    CONTROL(114,1,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_114.c || echo "restore of aes_114.c fails"
echo "x - extracting aes_12.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_12.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
```

```
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD appl_write(WORD ap_wid, WORD ap_wlength, char *ap_wbuff)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = ap_wid;
X    *a_tmp   = ap_wlength;
X    addr_in[0]= (LONG) ap_wbuff;
X    CONTROL(12,2,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_12.c || echo "restore of aes_12.c fails"
echo "x - extracting aes_120.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_120.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD shel_read(char *sh_rpcmd, char *sh_rptail)
X{
X    LONG *a_tmp = addr_in;
X    *a_tmp++ = (LONG) sh_rpcmd;
X    *a_tmp   = (LONG) sh_rptail;
```

```
X    CONTROL(120,0,1,2,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_120.c || echo "restore of aes_120.c fails"
echo "x - extracting aes_121.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_121.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD shel_write(WORD sh_wdoex, WORD sh_wisgr, WORD sh_wiscr, char *sh_wpcmd, char
*sh_wptail)
X{
X    WORD *a_tmp = int_in;
X    LONG *b_tmp = addr_in;
X    *a_tmp++ = sh_wdoex;
X    *a_tmp++ = sh_wisgr;
X    *a_tmp   = sh_wiscr;
X    *b_tmp++ = (LONG) sh_wpcmd;
X    *b_tmp   = (LONG) sh_wptail;
X    CONTROL(121,3,1,2,0);
X    AES_CALL();
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_121.c || echo "restore of aes_121.c fails"
echo "x - extracting aes_122.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_122.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
```

```
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD shel_get(char *sh_gaddr, WORD sh_glen)
X{
X    addr_in[0] = (LONG) sh_gaddr;
X    int_in[0]  = sh_glen;
X    CONTROL(122,1,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_122.c || echo "restore of aes_122.c fails"
echo "x - extracting aes_123.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_123.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD shel_put(char *sh_paddr, WORD sh_plen)
```

```
X{
X    addr_in[0] = (LONG) sh_paddr;
X    int_in[0]  = sh_plen;
X    CONTROL(123,1,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_123.c || echo "restore of aes_123.c fails"
echo "x - extracting aes_124.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_124.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD shel_find(char *sh_fpbuff)
X{
X    addr_in[0] = (LONG) sh_fpbuff;
X    CONTROL(124,0,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_124.c || echo "restore of aes_124.c fails"
echo "x - extracting aes_125.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_125.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
```

```
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD shel_envrn(char *sh_epvalue, char *sh_eparm)
X{
X    LONG *a_tmp = addr_in;
X    *a_tmp++ = (LONG) sh_epvalue;
X    *a_tmp   = (LONG) sh_eparm;
X    CONTROL(125,0,1,2,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_125.c || echo "restore of aes_125.c fails"
echo "x - extracting aes_13.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_13.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD appl_find( char *ap_fname)
X{
X    addr_in[0] = (LONG) ap_fname;
X    CONTROL(13,0,1,1,0);
X    AES_CALL();
```

```
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_13.c || echo "restore of aes_13.c fails"
echo "x - extracting aes_14.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_14.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD appl_tplay(char *ap_tpname, WORD ap_tpnum, WORD ap_tpscale)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = ap_tpnum;
X    *a_tmp   = ap_tpscale;
X    addr_in[0] = (LONG) ap_tpname;
X    CONTROL(14,2,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_14.c || echo "restore of aes_14.c fails"
echo "x - extracting aes_15.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_15.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
```

```
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD appl_trecord(char *ap_trmem, WORD ap_trcount)
X{
X    int_in[0] = ap_trcount;
X    addr_in[0] = (LONG) ap_trmem;
X    CONTROL(15,1,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_15.c || echo "restore of aes_15.c fails"
echo "x - extracting aes_19.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_19.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD appl_exit()
X{
X    CONTROL(19,0,1,0,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_19.c || echo "restore of aes_19.c fails"
```

```
echo "x - extracting aes_20.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_20.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD evnt_keybd()
X{
X    CONTROL(20,0,1,0,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_20.c || echo "restore of aes_20.c fails"
echo "x - extracting aes_21.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_21.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
```

```
X */
X#include "gem-guts.h"
X
XWORD evnt_button(WORD ev_bclicks, WORD ev_bmask, WORD ev_bstate, WORD *ev_bmx,
WORD *ev_bmy, WORD *ev_bbutton, WORD *ev_bkstate)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = ev_bclicks;
X    *a_tmp++ = ev_bmask;
X    *a_tmp   = ev_bstate;
X    CONTROL(21,3,5,0,0);
X    AES_CALL();
X    a_tmp = int_out;
X    *ev_bmx = *++a_tmp;
X    *ev_bmy = *++a_tmp;
X    *ev_bbutton = *++a_tmp;
X    *ev_bkstate = *++a_tmp;
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_21.c || echo "restore of aes_21.c fails"
echo "x - extracting aes_22.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_22.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD evnt_mouse(WORD ev_moflags, WORD ev_mox, WORD ev_moy, WORD ev_mowidth, WORD
ev_moheight, WORD *ev_momx, WORD *ev_momy, WORD *ev_mobutton, WORD *ev_mokstate)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = ev_moflags;
X    *a_tmp++ = ev_mox;
X    *a_tmp++ = ev_moy;
X    *a_tmp++ = ev_mowidth;
```

```
X    *a_tmp   = ev_moheight;
X    CONTROL(22,5,5,0,0);
X    AES_CALL();
X    a_tmp = int_out;
X    *ev_momx = *++a_tmp;
X    *ev_momy = *++a_tmp;
X    *ev_mobutton = *++a_tmp;
X    *ev_mokstate = *++a_tmp;
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_22.c || echo "restore of aes_22.c fails"
echo "x - extracting aes_23.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_23.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD evnt_mesag(char *ev_mgpbuff)
X{
X    addr_in[0] = (LONG) ev_mgpbuff;
X    CONTROL(23,0,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_23.c || echo "restore of aes_23.c fails"
echo "x - extracting aes_24.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_24.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
```

```
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD evnt_timer(unsigned WORD ev_tlocount, unsigned WORD ev_thicount)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = ev_tlocount;
X    *a_tmp   = ev_thicount;
X    CONTROL(24,2,1,0,0);
X    AES_CALL();
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_24.c || echo "restore of aes_24.c fails"
echo "x - extracting aes_25.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_25.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
```

```
X
XWORD evnt_multi( WORD a, WORD b, WORD c, WORD d, WORD e, WORD f, WORD g, WORD h,
WORD i, WORD j, WORD k, WORD l, WORD m, WORD n, char *buff, WORD o, WORD p, WORD
*q, WORD *r, WORD *s, WORD *t, WORD *u, WORD *v )
X{
X    WORD *tmp = int_in;
X    *tmp++ = a; *tmp++ = b; *tmp++ = c; *tmp++ = d; *tmp++ = e; *tmp++ = f;
X    *tmp++ = g; *tmp++ = h; *tmp++ = i; *tmp++ = j; *tmp++ = k; *tmp++ = l;
X    *tmp++ = m; *tmp++ = n; addr_in[0] = (LONG) buff; *tmp++ = o; *tmp   = p;
X    CONTROL(25,16,7,1,0);
X    AES_CALL();
X    tmp = int_out;
X    *q = *++tmp; *r = *++tmp; *s = *++tmp; *t = *++tmp; *u = *++tmp; *v = *++tmp;
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_25.c || echo "restore of aes_25.c fails"
echo "x - extracting aes_26.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_26.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD evnt_dclick(WORD ev_dnew, WORD ev_dgetset)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = ev_dnew;
X    *a_tmp   = ev_dgetset;
X    CONTROL(26,2,1,0,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_26.c || echo "restore of aes_26.c fails"
echo "x - extracting aes_30.c (Text)"
```

```
sed 's/^X//' << 'SHAR_EOF' > aes_30.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD menu_bar( OBJECT *me_btree, WORD me_bshow)
X{
X    addr_in[0] = (LONG) me_btree;
X    int_in[0]  = me_bshow;
X    CONTROL(30,1,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_30.c || echo "restore of aes_30.c fails"
echo "x - extracting aes_31.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_31.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
```

```
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD menu_icheck( OBJECT *me_ctree, WORD me_citem, WORD me_ccheck)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++  = me_citem;
X    *a_tmp    = me_ccheck;
X    addr_in[0] = (LONG) me_ctree;
X    CONTROL(31,2,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_31.c || echo "restore of aes_31.c fails"
echo "x - extracting aes_32.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_32.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD menu_ienable( OBJECT *me_etree, WORD me_eitem, WORD me_eenable)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++  = me_eitem;
X    *a_tmp    = me_eenable;
X    addr_in[0] = (LONG) me_etree;
X    CONTROL(32,2,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_32.c || echo "restore of aes_32.c fails"
```

```
echo "x - extracting aes_33.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_33.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD menu_tnormal( OBJECT *me_ntree, WORD me_ntitle, WORD me_nnormal)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++  = me_ntitle;
X    *a_tmp    = me_nnormal;
X    addr_in[0] = (LONG) me_ntree;
X    CONTROL(33,2,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_33.c || echo "restore of aes_33.c fails"
echo "x - extracting aes_34.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_34.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
```

```
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD menu_text( OBJECT *me_ttree, WORD me_titem, char *me_ttext)
X{
X    addr_in[0] = (LONG) me_ttree;
X    addr_in[1] = (LONG) me_ttext;
X    int_in[0]  = me_titem;
X    CONTROL(34,1,1,2,0);
X    AES_CALL();
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_34.c || echo "restore of aes_34.c fails"
echo "x - extracting aes_35.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_35.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD menu_register( WORD me_rapid, char *me_rpstring)
X{
X    int_in[0]  = me_rapid;
X    addr_in[0] = (LONG) me_rpstring;
X    CONTROL(35,1,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
X
```

```
SHAR_EOF
chmod 0644 aes_35.c || echo "restore of aes_35.c fails"
echo "x - extracting aes_40.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_40.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD objc_add( OBJECT *ob_atree, WORD ob_aparent, WORD ob_achild)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++  = ob_aparent;
X    *a_tmp    = ob_achild;
X    addr_in[0] = (LONG) ob_atree;
X    CONTROL(40,2,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_40.c || echo "restore of aes_40.c fails"
echo "x - extracting aes_41.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_41.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
```

```
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD objc_delete( OBJECT *ob_dltree, WORD ob_dlobject)
X{
X    addr_in[0] = (LONG) ob_dltree;
X    int_in[0]  = ob_dlobject;
X    CONTROL(41,1,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
X
X
SHAR_EOF
chmod 0644 aes_41.c || echo "restore of aes_41.c fails"
echo "x - extracting aes_42.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_42.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD objc_draw( OBJECT *ob_drtree, WORD ob_drstartob, WORD ob_drdepth, WORD
ob_drxclip, WORD ob_dryclip, WORD ob_drwclip, WORD ob_drhclip)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++  = ob_drstartob;
X    *a_tmp++  = ob_drdepth;
X    *a_tmp++  = ob_drxclip;
```

```
X    *a_tmp++  = ob_dryclip;
X    *a_tmp++  = ob_drwclip;
X    *a_tmp    = ob_drhclip;
X    addr_in[0] = (LONG) ob_drtree;
X    CONTROL(42,6,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
X
X
SHAR_EOF
chmod 0644 aes_42.c || echo "restore of aes_42.c fails"
echo "x - extracting aes_43.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_43.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD objc_find( OBJECT *ob_ftree, WORD ob_fstartob, WORD ob_fdepth, WORD ob_fmx,
WORD ob_fmy)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = ob_fstartob;
X    *a_tmp++ = ob_fdepth;
X    *a_tmp++ = ob_fmx;
X    *a_tmp   = ob_fmy;
X    addr_in[0] = (LONG) ob_ftree;
X    CONTROL(43,4,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
X
X
X
SHAR_EOF
```

```
chmod 0644 aes_43.c || echo "restore of aes_43.c fails"
echo "x - extracting aes_44.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_44.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD objc_offset(OBJECT *ob_oftree, WORD ob_ofobject, WORD *ob_ofxoff, WORD
*ob_ofyoff)
X{
X    WORD *a_tmp = int_out;
X    addr_in[0] = (LONG) ob_oftree;
X    int_in[0]  = ob_ofobject;
X    CONTROL(44,1,3,1,0);
X    *ob_ofxoff = *++a_tmp;
X    *ob_ofyoff = *++a_tmp;
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_44.c || echo "restore of aes_44.c fails"
echo "x - extracting aes_45.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_45.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
```

```
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD objc_order( OBJECT *ob_ortree, WORD ob_orobject, WORD ob_ornewpos)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = ob_orobject;
X    *a_tmp   = ob_ornewpos;
X    addr_in[0] = (LONG) ob_ortree;
X    CONTROL(45,2,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
X
X
X
SHAR_EOF
chmod 0644 aes_45.c || echo "restore of aes_45.c fails"
echo "x - extracting aes_46.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_46.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD objc_edit( OBJECT *ob_edtree, WORD ob_edobject, WORD ob_edchar, WORD
*ob_edidx, WORD ob_edkind)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = ob_edobject;
```

```
X    *a_tmp++ = ob_edchar;
X    *a_tmp++ = *ob_edidx;
X    *a_tmp   = ob_edkind;
X    addr_in[0] = (LONG) ob_edtree;
X    CONTROL(46,4,2,1,0);
X    AES_CALL();
X    *ob_edidx = int_out[1];
X    return int_out[0];
X}
X
X
X
SHAR_EOF
chmod 0644 aes_46.c || echo "restore of aes_46.c fails"
echo "x - extracting aes_47.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_47.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD objc_change( OBJECT *ob_ctree, WORD ob_cobject, WORD ob_cresvd, WORD
ob_cxclip, WORD ob_cyclip, WORD ob_cwclip, WORD ob_chclip, WORD ob_cnewstate, WORD
ob_credraw)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = ob_cobject;
X    *a_tmp++ = ob_cresvd;
X    *a_tmp++ = ob_cxclip;
X    *a_tmp++ = ob_cyclip;
X    *a_tmp++ = ob_cwclip;
X    *a_tmp++ = ob_chclip;
X    *a_tmp++ = ob_cnewstate;
X    *a_tmp   = ob_credraw;
X    addr_in[0] = (LONG) ob_ctree;
X    CONTROL(43,4,1,1,0);
```

```
X    AES_CALL();
X    return int_out[0];
X}
X
X
X
SHAR_EOF
chmod 0644 aes_47.c || echo "restore of aes_47.c fails"
echo "x - extracting aes_50.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_50.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD form_do(OBJECT *fo_dotree, WORD fo_dostartob)
X{
X    addr_in[0] = (LONG) fo_dotree;
X    int_in[0]  = fo_dostartob;
X    CONTROL(50,1,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_50.c || echo "restore of aes_50.c fails"
echo "x - extracting aes_51.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_51.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
```

```
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
SHAR_EOF
echo "End of part 2"
echo "File aes_51.c is continued in part 3"
echo "3" > s2_seq_.tmp
exit 0


From -715280355625776727
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,afd75fff35a2e4fe,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-20 16:30:22 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!ames!apple!sun-barr!newstop!sun!
grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i058: gccgem -- GEM libraries for GCC  part03/06
Keywords: shar, C
Message-ID: <33906@grapevine.uucp>
Date: 20 Jul 89 23:30:22 GMT
Sender: news@grapevine.uucp
Lines: 1353
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: uunet!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 2, Issue 58
Archive-name: gccgem/part03

#!/bin/sh
# this is part 3 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file aes_51.c continued
#
CurArch=3
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file aes_51.c"
```

```
sed 's/^X//' << 'SHAR_EOF' >> aes_51.c
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD form_dial(WORD fo_diflag, WORD fo_dilittlx, WORD fo_dilittly, WORD
fo_dilittlw, WORD fo_dilittlh, WORD fo_dibigx, WORD fo_dibigy, WORD fo_dibigw,
WORD fo_dibigh)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = fo_diflag;
X    *a_tmp++ = fo_dilittlx;
X    *a_tmp++ = fo_dilittly;
X    *a_tmp++ = fo_dilittlw;
X    *a_tmp++ = fo_dilittlh;
X    *a_tmp++ = fo_dibigx;
X    *a_tmp++ = fo_dibigy;
X    *a_tmp++ = fo_dibigw;
X    *a_tmp   = fo_dibigh;
X    CONTROL(51,9,1,0,0);
X    AES_CALL();
X    return int_out[0];
X}
X
SHAR_EOF
echo "File aes_51.c is complete"
chmod 0644 aes_51.c || echo "restore of aes_51.c fails"
echo "x - extracting aes_52.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_52.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD form_alert(WORD fo_adefbttn, char *fo_astring)
X{
```

```
X    int_in[0] = fo_adefbttn;
X    addr_in[0] = (LONG) fo_astring;
X    CONTROL(52,1,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_52.c || echo "restore of aes_52.c fails"
echo "x - extracting aes_53.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_53.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD form_error(WORD fo_enum)
X{
X    int_in[0] = fo_enum;
X    CONTROL(53,1,1,0,0);
X    AES_CALL();
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_53.c || echo "restore of aes_53.c fails"
echo "x - extracting aes_54.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_54.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
```

```
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD form_center(OBJECT *fo_ctree, WORD *fo_cx, WORD *fo_cy, WORD *fo_cw, WORD
*fo_ch)
X{
X    WORD *a_tmp = int_out;
X    addr_in[0] = (LONG) fo_ctree;
X    CONTROL(54,0,5,1,0);
X    AES_CALL();
X    *fo_cx = *++a_tmp;
X    *fo_cy = *++a_tmp;
X    *fo_cw = *++a_tmp;
X    *fo_ch = *++a_tmp;
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_54.c || echo "restore of aes_54.c fails"
echo "x - extracting aes_55.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_55.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
```

```
X
XWORD form_keybd(OBJECT *fo_ktree, WORD fo_kobject, WORD fo_kobnext, WORD
fo_kchar, WORD *fo_knxtobject, WORD *fo_knxtchar)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = fo_kobject;
X    *a_tmp++ = fo_kobnext;
X    *a_tmp   = fo_kchar;
X    addr_in[0] = (LONG) fo_ktree;
X    CONTROL(55,3,3,1,0);
X    AES_CALL();
X    a_tmp = int_out;
X    *fo_knxtobject = *++a_tmp;
X    *fo_knxtchar   = *++a_tmp;
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_55.c || echo "restore of aes_55.c fails"
echo "x - extracting aes_56.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_56.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD form_button(OBJECT *fo_btree, WORD fo_bobject, WORD fo_bclicks, WORD
*fo_bnxtobj)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = fo_bobject;
X    *a_tmp   = fo_bclicks;
X    addr_in[0] = (LONG) fo_btree;
X    CONTROL(56,2,2,1,0);
X    AES_CALL();
X    *fo_bnxtobj = int_out[1];
```

```
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_56.c || echo "restore of aes_56.c fails"
echo "x - extracting aes_70.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_70.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD graf_rubberbox(WORD gr_rx, WORD gr_ry, WORD gr_rminwidth, WORD
gr_rminheight, WORD *gr_rlastwidth, WORD *gr_rlastheight)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = gr_rx;
X    *a_tmp++ = gr_ry;
X    *a_tmp++ = gr_rminwidth;
X    *a_tmp   = gr_rminheight;
X    CONTROL(70,4,3,0,0);
X    AES_CALL();
X    a_tmp = int_out;
X    *gr_rlastwidth  = *++a_tmp;
X    *gr_rlastheight = *++a_tmp;
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_70.c || echo "restore of aes_70.c fails"
echo "x - extracting aes_71.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_71.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
```

```
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD graf_dragbox( WORD gr_dwidth, WORD gr_dheight, WORD gr_dstartx, WORD
gr_dstarty, WORD gr_dboundx, WORD gr_dboundy, WORD gr_dboundw, WORD gr_dboundh,
WORD *gr_dfinishx, WORD *gr_dfinishy)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = gr_dwidth;
X    *a_tmp++ = gr_dheight;
X    *a_tmp++ = gr_dstartx;
X    *a_tmp++ = gr_dstarty;
X    *a_tmp++ = gr_dboundx;
X    *a_tmp++ = gr_dboundy;
X    *a_tmp++ = gr_dboundw;
X    *a_tmp   = gr_dboundh;
X    CONTROL(71,8,3,0,0);
X    AES_CALL();
X    a_tmp = int_out;
X    *gr_dfinishx = *++a_tmp;
X    *gr_dfinishy = *++a_tmp;
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_71.c || echo "restore of aes_71.c fails"
echo "x - extracting aes_72.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_72.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
```

```
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD graf_movebox( WORD gr_mwidth, WORD gr_mheight, WORD gr_msourcex, WORD
gr_msourcey, WORD gr_mdestx, WORD gr_mdesty)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = gr_mwidth;
X    *a_tmp++ = gr_mheight;
X    *a_tmp++ = gr_msourcex;
X    *a_tmp++ = gr_msourcey;
X    *a_tmp++ = gr_mdestx;
X    *a_tmp   = gr_mdesty;
X    CONTROL(72,6,1,0,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_72.c || echo "restore of aes_72.c fails"
echo "x - extracting aes_73.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_73.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD graf_growbox( WORD gr_gstx, WORD gr_gsty, WORD gr_gstwidth, WORD
gr_gstheight, WORD gr_gfinx, WORD gr_gfiny, WORD gr_gfinwidth, WORD gr_gfinheight)
X{
```

```
X    WORD *a_tmp = int_in;
X    *a_tmp++ = gr_gstx;
X    *a_tmp++ = gr_gsty;
X    *a_tmp++ = gr_gstwidth;
X    *a_tmp++ = gr_gstheight;
X    *a_tmp++ = gr_gfinx;
X    *a_tmp++ = gr_gfiny;
X    *a_tmp++ = gr_gfinwidth;
X    *a_tmp   = gr_gfinheight;
X    CONTROL(73,8,1,0,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_73.c || echo "restore of aes_73.c fails"
echo "x - extracting aes_74.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_74.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD graf_shrinkbox( WORD gr_sfinx, WORD gr_sfiny, WORD gr_sfinwidth, WORD
gr_sfinheight, WORD gr_sstx, WORD gr_ssty, WORD gr_sstwidth, WORD gr_sstheight)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = gr_sfinx;
X    *a_tmp++ = gr_sfiny;
X    *a_tmp++ = gr_sfinwidth;
X    *a_tmp++ = gr_sfinheight;
X    *a_tmp++ = gr_sstx;
X    *a_tmp++ = gr_ssty;
X    *a_tmp++ = gr_sstwidth;
X    *a_tmp   = gr_sstheight;
X    CONTROL(74,8,1,0,0);
X    AES_CALL();
```

```
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_74.c || echo "restore of aes_74.c fails"
echo "x - extracting aes_75.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_75.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD graf_watchbox( OBJECT *gr_wptree, WORD gr_wobject, WORD gr_winstate, WORD
gr_woutstate)
X{
X    WORD *a_tmp = int_in;
X    *++a_tmp = gr_wobject;
X    *++a_tmp = gr_winstate;
X    *++a_tmp = gr_woutstate;
X    addr_in[0] = (LONG) gr_wptree;
X    CONTROL(75,4,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_75.c || echo "restore of aes_75.c fails"
echo "x - extracting aes_76.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_76.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
```

```
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD graf_slidebox( OBJECT *gr_slptree, WORD gr_slparent, WORD gr_slobject, WORD
gr_slvh)
X{
X    WORD *a_tmp = int_in;
X    *a_tmp++ = gr_slparent;
X    *a_tmp++ = gr_slobject;
X    *a_tmp   = gr_slvh;
X    addr_in[0] = (LONG) gr_slptree;
X    CONTROL(76,3,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_76.c || echo "restore of aes_76.c fails"
echo "x - extracting aes_77.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_77.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD graf_handle(WORD *gr_hwchar, WORD *gr_hhchar, WORD *gr_hwbox, WORD
```

```
*gr_hhbox)
X{
X    WORD *a_tmp = int_out;
X    CONTROL(77,0,5,0,0);
X    AES_CALL();
X    *gr_hwchar = *++a_tmp;
X    *gr_hhchar = *++a_tmp;
X    *gr_hwbox  = *++a_tmp;
X    *gr_hhbox  = *++a_tmp;
X    return int_out[0];
X}
X
SHAR_EOF
chmod 0644 aes_77.c || echo "restore of aes_77.c fails"
echo "x - extracting aes_78.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_78.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD graf_mouse(WORD gr_monumber, MFORM *gr_mofaddr)
X{
X    int_in[0] = gr_monumber;
X    addr_in[0]= (LONG) gr_mofaddr;
X    CONTROL(78,1,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_78.c || echo "restore of aes_78.c fails"
echo "x - extracting aes_79.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_79.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
```

```
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD graf_mkstate( WORD *gr_mkmx, WORD *gr_mkmy, WORD *gr_mkmstate, WORD
*gr_mkkstate)
X{
X    WORD *a_tmp = int_out;
X    CONTROL(79,0,5,0,0);
X    AES_CALL();
X    *gr_mkmx = *++a_tmp;
X    *gr_mkmy = *++a_tmp;
X    *gr_mkmstate = *++a_tmp;
X    *gr_mkkstate = *++a_tmp;
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_79.c || echo "restore of aes_79.c fails"
echo "x - extracting aes_80.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_80.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
```

```
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD scrp_read( char *sc_rpscrap)
X{
X    addr_in[0] = (LONG) sc_rpscrap;
X    CONTROL(80,0,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_80.c || echo "restore of aes_80.c fails"
echo "x - extracting aes_81.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_81.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD scrp_write( char *sc_wpscrap)
X{
X    addr_in[0] = (LONG) sc_wpscrap;
X    CONTROL(81,0,1,1,0);
X    AES_CALL();
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_81.c || echo "restore of aes_81.c fails"
echo "x - extracting aes_90.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_90.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
```

```
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD fsel_input(char *fs_iinpath, char *fs_iinsel, WORD *fs_iexbutton)
X{
X    LONG *a_tmp = addr_in;
X    *a_tmp++ = (LONG) fs_iinpath;
X    *a_tmp   = (LONG) fs_iinsel;
X    CONTROL(90,0,2,2,0);
X    AES_CALL();
X    *fs_iexbutton = int_out[1];
X    return int_out[0];
X}
SHAR_EOF
chmod 0644 aes_90.c || echo "restore of aes_90.c fails"
echo "x - extracting aes_91.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > aes_91.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include <osbind.h>
X#include "gem-guts.h"
```

```
X
Xstatic int old_tosversion = FALSE;
X
Xstatic VOID check_tos()
X{
X   short ** _sys_base = (short **) 0x000004f2L;
X   if (*(*_sys_base+1) < 0x0104) old_tosversion = TRUE;
X}
X
XWORD fsel_exinput(char *fs_iinpath, char *fs_iinsel, WORD *fs_iexbutton, char
*fs_label)
X{
X    LONG *a_tmp = addr_in;
X    VOID check_tos();
X
X    Supexec(check_tos);
X    if (old_tosversion)
X         return fsel_input(fs_iinpath, fs_iinsel, fs_iexbutton);
X    else {
X         *a_tmp++ = (LONG) fs_iinpath;
X         *a_tmp++ = (LONG) fs_iinsel;
X         *a_tmp   = (LONG) fs_label;
X         CONTROL(91,0,2,3,0);
X         AES_CALL();
X         *fs_iexbutton = int_out[1];
X         return int_out[0];
X    }
X}
SHAR_EOF
chmod 0644 aes_91.c || echo "restore of aes_91.c fails"
echo "x - extracting felder.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > felder.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include <portab.h>
```

```
X
XWORD contrl[128];
XWORD intin[128];
XWORD intout[128];
XWORD ptsin[128];
XWORD ptsout[128];
X
XLONG pblock[5];
X
XWORD global[15];
XWORD control[5];
XWORD int_in[16];
XWORD int_out[7];
XLONG addr_in[3];
XLONG addr_out[1];
SHAR_EOF
chmod 0644 felder.c || echo "restore of felder.c fails"
echo "x - extracting gem-guts.h (Text)"
sed 's/^X//' << 'SHAR_EOF' > gem-guts.h &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include <portab.h>
X
X/*
X *  VDI
X */
X
Xextern WORD contrl[];
Xextern WORD intin[];
Xextern WORD intout[];
Xextern WORD ptsin[];
Xextern WORD ptsout[];
X
Xextern WORD *pblock[];
X
```

```
X#define VDI_CALL()    { asm("movеml a0-a1,sp@-"); asm("movel  #_pblock,d1");
asm("movew    #115,d0"); asm("trap   #2"); asm("moveml sp@+,a0-a1"); }
X
XVOID vdi_macro_a(WORD nummer, WORD handle);
XWORD vdi_macro_b(WORD nummer, WORD eingabe1, WORD eingabe2);
XVOID vdi_macro_c(WORD nummer, WORD eingabe1, WORD eingabe2, WORD eingabe3[]);
XVOID vdi_macro_d(WORD nummer, WORD handle, WORD eingabe);
X
Xtypedef struct {
X        char *fd_addr;
X        WORD fd_w;
X        WORD fd_h;
X        WORD fd_wdwidth;
X        WORD fd_stand;
X        WORD fd_nplanes;
X        WORD fd_r1;
X        WORD fd_r2;
X        WORD fd_r3;
X    } FDB;
X
X/*
X *  AES
X */
X
Xextern WORD global[];
Xextern WORD control[];
Xextern WORD int_in[];
Xextern WORD int_out[];
Xextern LONG addr_in[];
Xextern LONG addr_out[];
X
Xtypedef struct {
X    WORD *cb_pcontrol;
X    WORD *cb_pglobal;
X    WORD *cb_pintin;
X    WORD *cb_pintout;
X    LONG *cb_padrin;
X    LONG *cb_padrout;
X    } AESPB;
X
Xextern WORD ge_apid;
Xextern AESPB aesbp;
X
X#define CONTROL(_a,_b,_c,_d,_e) { WORD *c_tmp = control; *c_tmp++ = _a; *c_tmp++
= _b; *c_tmp++ = _c; *c_tmp = _d; /* control[4] ist immer 0 - appl_init() */ }
X
X#define AES_CALL() { asm("movеml    a0-a3,sp@-"); asm("movel
     #_aespb,d1"); asm("movew    #200,d0"); asm("trap   #2"); asm("moveml sp@+,a0-
a3"); }
X
Xtypedef struct {
X        WORD ob_next;
X        WORD ob_head;
```

```
X         WORD ob_tail;
X         unsigned WORD      ob_type;
X         unsigned WORD      ob_flags;
X         unsigned WORD      ob_state;
X         char *ob_spec;
X         WORD ob_x;
X         WORD ob_y;
X         WORD ob_width;
X         WORD ob_height;
X    } OBJECT;
X
Xtypedef struct {
X         char *te_ptext;
X         char *te_ptmplt;
X         char *te_pvalid;
X         WORD te_font;
X         WORD te_resvd1;
X         WORD te_just;
X         WORD te_color;
X         WORD te_resvd2;
X         WORD te_thickness;
X         WORD te_txtlen;
X         WORD te_tmplen;
X    } TEDINFO;
X
Xtypedef struct {
X         WORD *ib_pmask;
X         WORD *ib_pdata;
X         char *ib_ptext;
X         WORD ib_char;
X         WORD ib_xchar;
X         WORD ib_ychar;
X         WORD ib_xicon;
X         WORD ib_yicon;
X         WORD ib_wicon;
X         WORD ib_hicon;
X         WORD ib_xtext;
X         WORD ib_ytext;
X         WORD ib_wtext;
X         WORD ib_htext;
X         WORD ib_resvd;
X    } ICONBLK;
X
Xtypedef struct {
X         WORD *bi_pdata;
X         WORD bi_wb;
X         WORD bi_hl;
X         WORD bi_x;
X         WORD bi_y;
X         WORD bi_color;
X    } BITBLK;
X
Xtypedef struct {
```

```
X         WORD (*ub_code)();
X         LONG ub_parm;
X    } USERBLK;
X
Xtypedef struct {
X         OBJECT   *pb_tree;
X         WORD pb_obj;
X         WORD pb_prevstate;
X         WORD pb_currstate;
X         WORD pb_x;
X         WORD pb_y;
X         WORD pb_w;
X         WORD pb_h;
X         WORD pb_xc;
X         WORD pb_yc;
X         WORD pb_wc;
X         WORD pb_hc;
X         LONG pb_parm;
X    } PARMBLK;
X
Xtypedef struct {
X         WORD mf_xhot;
X         WORD mf_yhot;
X         WORD mf_nplanes;
X         WORD mf_fg;
X         WORD mf_bg;
X         WORD mf_mask[16];
X         WORD mf_data[16];
X    } MFORM;
SHAR_EOF
chmod 0644 gem-guts.h || echo "restore of gem-guts.h fails"
echo "x - extracting vdi.h (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi.h &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
```

```
X/*
X *    VDI.H - Headerfile fuer die VDI Bindings
X *
X *    version 1.0    13.7.89  stefan brueck (sb)
X */
X
X#ifndef __VDI_H
X#define __VDI_H
X
X#include <portab.h>
X
Xextern WORD contrl[];
Xextern WORD intin[];
Xextern WORD intout[];
Xextern WORD ptsin[];
Xextern WORD ptsout[];
X
Xextern WORD *pblock[];
X
X/*
X * die einzelnen strukturen
X */
X
Xtypedef struct {
X      char *fd_addr;
X      WORD fd_w;
X      WORD fd_h;
X      WORD fd_wdwidth;
X      WORD fd_stand;
X      WORD fd_nplanes;
X      WORD fd_r1;
X      WORD fd_r2;
X      WORD fd_r3;
X   } FDB;
X
X/*
X * die einzelnen funtionen
X */
X
X/****************** kontrollfunktionen ***********************/
X
XVOID  v_opnwk(WORD work_in[], WORD *handle, WORD work_out[]);
XVOID  v_clswk(WORD handle);
XVOID  v_opnvwk(WORD work_in[], WORD *handle, WORD work_out[]);
XVOID  v_clsvwk(WORD handle);
XVOID  v_clrwk(WORD handle);
XVOID  v_updwk(WORD handle);
XWORD  vst_load_fonts(WORD handle, WORD select);
XVOID  vst_unload_fonts(WORD handle, WORD select);
XVOID  vs_clip(WORD handle, WORD clip_flag, WORD pxarray[]);
X
X/****************** ausgabefunktionen ***********************/
X
```

```
XVOID  v_pline(WORD handle, WORD count, WORD pxyarray[]);
XVOID  v_pmarker(WORD handle, WORD count, WORD pxyarray[]);
XVOID  v_gtext(WORD handle, WORD x, WORD y, char *string);
XVOID  v_fillarea(WORD handle, WORD count, WORD pxyarray[]);
XVOID  v_cellarray(WORD handle, WORD pxyarray[], WORD row_length, WORD el_used,
WORD num_rows, WORD wrt_mode, WORD colarray[]);
XVOID  v_contourfill(WORD handle, WORD x, WORD y, WORD index);
XVOID  vr_recfl(WORD handle, WORD pxyarray[]);
XVOID  v_bar(WORD handle, WORD pxyarray[]);
XVOID  v_arc(WORD handle, WORD x, WORD y, WORD radius, WORD begang, WORD endang);
XVOID  v_pie(WORD handle, WORD x, WORD y, WORD radius, WORD begang, WORD endang);
XVOID  v_circle(WORD handle, WORD x, WORD y, WORD radius);
XVOID  v_ellipse(WORD handle, WORD x, WORD y, WORD xradius, WORD yradius);
XVOID  v_ellarc(WORD handle, WORD x, WORD y, WORD xradius, WORD yradius, WORD
begang, WORD endang);
XVOID  v_ellpie(WORD handle, WORD x, WORD y, WORD xradius, WORD yradius, WORD
begang, WORD endang);
XVOID  v_rbox(WORD handle, WORD xyarray[]);
XVOID  v_rfbox(WORD handle, WORD xyarray[]);
XVOID  v_justified(WORD handle, WORD x, WORD y, char *string, WORD word_space,
WORD char_space);
X
X/***************** attributfunktionen *****************/
X
XWORD  vswr_mode(WORD handle, WORD mode);
XVOID  vs_color(WORD handle, WORD index, WORD rgb_in[]);
XWORD  vsl_type(WORD handle, WORD style);
XVOID  vsl_udsty(WORD handle, WORD pattern);
XWORD  vsl_width(WORD handle, WORD width);
XWORD  vsl_color(WORD handle, WORD color_index);
XVOID  vsl_ends(WORD handle, WORD beg_style, WORD end_style);
XWORD  vsm_type(WORD handle, WORD symbol);
XWORD  vsm_height(WORD handle, WORD height);
XWORD  vsm_color(WORD handle, WORD color_index);
XVOID  vst_height(WORD handle, WORD height, WORD *char_width, WORD *char_height,
WORD *cell_width, WORD *cell_height);
XWORD  vst_point(WORD handle, WORD point, WORD *char_width, WORD *char_height,
WORD *cell_width, WORD *cell_height);
XWORD  vst_rotation(WORD handle, WORD angle);
XWORD  vst_font(WORD handle, WORD font);
XWORD  vst_color(WORD handle, WORD color_index);
XWORD  vst_effects(WORD handle, WORD effect);
XVOID  vst_alignment(WORD handle, WORD hor_in, WORD vert_in, WORD *hor_out, WORD
*vert_out);
XWORD  vsf_interior(WORD handle, WORD style);
XWORD    vsf_style(WORD handle, WORD style_index);
XWORD  vsf_color(WORD handle, WORD color_index);
XWORD  vsf_perimeter(WORD handle, WORD per_vis);
XVOID  vsf_updat(WORD handle, WORD pfill_pat[], WORD planes);
X
X/******************** rasteroperationen ******************/
X
XVOID  vro_cpyfm(WORD handle, WORD wr_mode, WORD pxyarray[], FDB *psrcMFDB, FDB
```

```
*pdesMFDB);
XVOID  vrt_cpyfm(WORD handle, WORD wr_mode, WORD pxyarray[], FDB *psrcMFDB, FDB
*pdesMFDB, WORD color_index[]);
XVOID  vr_trnfm(WORD handle, FDB *psrcMFDB, FDB *pdesMFDB);
XVOID  v_get_pixel(WORD handle, WORD x, WORD y, WORD *pel, WORD *index);
X
X/******************** eingabefunktionen ******************/
X
XWORD  vsin_mode(WORD handle, WORD dev_type, WORD mode);
XVOID  vrq_locator(WORD handle, WORD x, WORD y, WORD *xout, WORD *yout, WORD
*term);
XWORD  vsm_locator(WORD handle, WORD x, WORD y, WORD *xout, WORD *yout, WORD
*term);
XVOID  vrq_valuator(WORD handle, WORD valuator_in, WORD *valuator_out, WORD
*terminator);
XVOID  vsm_valuator(WORD handle, WORD valuator_in, WORD *valuator_out, WORD
*terminator, WORD *status);
XVOID  vrq_choice(WORD handle, WORD ch_in, WORD *ch_out);
XWORD  vsm_choice(WORD handle, WORD *choice);
XVOID  vrq_string(WORD handle, WORD max_length, WORD echo_mode, WORD echo_xy[],
char *string);
XWORD  vsm_string(WORD handle, WORD max_length, WORD echo_mode, WORD echo_xy[],
char *string);
XVOID  vsc_form(WORD handle, WORD pcur_form[]);
XVOID  vex_timv(WORD handle, LONG tim_addr, LONG *otim_addr, WORD *tim_conv);
XWORD  v_show_c(WORD handle, WORD reset);
XWORD  v_hide_c(WORD handle);
XVOID  vq_mouse(WORD handle, WORD *pstatus, WORD *x, WORD *y);
XVOID  vex_butv(WORD handle, LONG pusrcode, LONG *psavcode);
XVOID  vex_motv(WORD handle, LONG pusrcode, LONG *psavcode);
XVOID  vex_curv(WORD handle, LONG pusrcode, LONG *psavcode);
XWORD  vq_key_s(WORD handle, WORD *pstatus);
X
X/******************** auskunftsfunktionen ********************/
X
XVOID  vq_extnd(WORD handle, WORD owflag, WORD work_out[]);
XVOID  vq_color(WORD handle, WORD color_index, WORD set_flag, WORD rgb[]);
XVOID  vql_attributes( WORD handle, WORD attrib[]);
XVOID  vqm_attributes( WORD handle, WORD attrib[]);
XVOID  vqf_attributes( WORD handle, WORD attrib[]);
XVOID  vqt_attributes( WORD handle, WORD attrib[]);
XVOID  vqt_extent( WORD handle, char *string, WORD extent[]);
XWORD  vqt_width( WORD handle, WORD character, WORD *cell_width, WORD *left_delta,
WORD *right_delta);
XWORD  vqt_name( WORD handle, WORD element_num, WORD name[]);
XVOID  vq_cellarray( WORD handle, WORD pxyarray[], WORD row_length, WORD num_rows,
WORD *el_used, WORD *rows_used, WORD *status, WORD colarray[]);
XVOID  vqin_mode( WORD handle, WORD dev_type, WORD *input_mode);
XVOID  vqt_fontinfo( WORD handle, WORD *minADE, WORD *maxADE, WORD distances[],
WORD *maxwidth, WORD effects[]);
XVOID  vq_chcells( WORD handle, WORD *rows, WORD *colums);
XVOID  v_exit_cur( WORD handle);
XVOID  v_enter_cur( WORD handle);
```

```
XVOID  v_curup( WORD handle);
XVOID  v_curdown( WORD handle);
XVOID  v_curright( WORD handle);
XVOID  v_curleft( WORD handle);
XVOID  v_curhome( WORD handle);
XVOID  v_eeos( WORD handle);
XVOID  v_eeol( WORD handle);
XVOID  v_curaddress( WORD handle, WORD row, WORD column);
XVOID  v_curtext( WORD handle, char *string);
XVOID  v_rvon( WORD handle);
XVOID  v_rvoff( WORD handle);
XVOID  vq_curaddress( WORD handle, WORD *row, WORD *column);
XWORD  vq_tabstatus( WORD handle);
XVOID  v_hardcopy( WORD handle);
XVOID  v_dspcur( WORD handle, WORD x, WORD y);
XVOID  v_rmcur( WORD handle);
XVOID  v_form_adv( WORD handle);
XVOID  v_output_window( WORD handle, WORD xyarray[]);
XVOID  v_clear_disp_list( WORD handle);
XVOID  v_bit_image( WORD handle, char *filename, WORD aspect, WORD scaling, WORD
num_pts, WORD xyarray[]);
X
X/*
X * not yet implemented!
X   VOID  vs_palette( WORD handle, WORD palette);
X   VOID  vqp_films( WORD handle, char film_names[]);
X   VOID  vqp_state( WORD handle, WORD *port, WORD *film_name, WORD *lightness,
WORD *interlace, WORD *planes, WORD *indexes);
X   VOID  vsp_state( WORD handle, WORD port, WORD film_name, WORD lightness, WORD
interlace, WORD planes, WORD indexes[]);
X   VOID  vsp_save( WORD handle);
X   VOID  vsp_message( WORD handle);
X   VOID  vsp_error( WORD handle);
X*/
XVOID  v_meta_extents( WORD handle, WORD min_x, WORD min_y, WORD max_x, WORD
max_y);
XVOID  v_write_meta( WORD handle, WORD num_intin, WORD a_intin[], WORD num_ptsin,
WORD a_ptsin[]);
XVOID  vm_filename( WORD handle, char *filename);
XVOID  v_offset( WORD handle, WORD offset);
XVOID  v_fontinit( WORD handle, WORD fh_high, WORD fh_low);
X
X#endif /* not __VDI_H */
SHAR_EOF
chmod 0644 vdi.h || echo "restore of vdi.h fails"
echo "x - extracting vdi11_10.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi11_10.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
```

```
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_justified(WORD handle, WORD x, WORD y, char *string, WORD length, WORD
word_space, WORD char_space)
X{
X    WORD *tmp;
X
X    ptsin[0] = x;
X    ptsin[1] = y;
X    ptsin[2] = length;
X    ptsin[3] = 0;
X    intin[0] = word_space;
X    intin[1] = char_space;
X    tmp = &intin[2];
X    while(*tmp++ = *string++);
X    contrl[0] = 11;
X    contrl[1] = 2;
X    contrl[3] = strlen(string);
X    contrl[5] = 10;
X    contrl[6] = handle;
X    VDI_CALL();
X}
X
SHAR_EOF
chmod 0644 vdi11_10.c || echo "restore of vdi11_10.c fails"
echo "x - extracting vdi5_1.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_1.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
```

```
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vq_chcells( WORD handle, WORD *rows, WORD *columns)
X{
X    contrl[5] = 1;
X    vdi_macro_a(5, handle);
X    *rows = intout[0];
X    *columns = intout[1];
X}
SHAR_EOF
chmod 0644 vdi5_1.c || echo "restore of vdi5_1.c fails"
echo "x - extracting vdi5_10.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_10.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_eeol( WORD handle)
X{
X    contrl[5] = 10;
X    vdi_macro_a(5, handle);
X}
X
X
SHAR_EOF
chmod 0644 vdi5_10.c || echo "restore of vdi5_10.c fails"
echo "x - extracting vdi5_100.c (Text)"
```

```
sed 's/^X//' << 'SHAR_EOF' > vdi5_100.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vm_filename( WORD handle, char *filename)
X{
X    int i = 0;
X    while (intin[i++] = *filename++);
X    contrl[0] = 5;
X    contrl[1] = 0;
X    contrl[3] = --i;
X    contrl[5] = 100;
X    contrl[6] = handle;
X    VDI_CALL();
X}
SHAR_EOF
chmod 0644 vdi5_100.c || echo "restore of vdi5_100.c fails"
echo "x - extracting vdi5_101.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_101.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
SHAR_EOF
echo "End of part 3"
echo "File vdi5_101.c is continued in part 4"
echo "4" > s2_seq_.tmp
exit 0
```

From -6369395920457875242
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,a444c5aa4f242002,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-20 16:30:31 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!rutgers!apple!sun-barr!newstop!sun!
grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i059: gccgem -- GEM libraries for GCC part04/06
Keywords: shar, C
Message-ID: <33907@grapevine.uucp>
Date: 20 Jul 89 23:30:31 GMT
Sender: news@grapevine.uucp
Lines: 1356
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: uunet!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 2, Issue 59
Archive-name: gccgem/part04

```
#!/bin/sh
# this is part 4 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file vdi5_101.c continued
#
CurArch=4
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file vdi5_101.c"
sed 's/^X//' << 'SHAR_EOF' >> vdi5_101.c
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_offset( WORD handle, WORD offset)
```

```
X{
X    contrl[5] = 101;
X    vdi_macro_d(5, handle, offset);
X}
SHAR_EOF
echo "File vdi5_101.c is complete"
chmod 0644 vdi5_101.c || echo "restore of vdi5_101.c fails"
echo "x - extracting vdi5_102.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_102.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_fontinit( WORD handle, WORD fh_high, WORD fh_low)
X{
X    intin[0] = fh_high;
X    intin[1] = fh_low;
X    contrl[0] = 5;
X    contrl[1] = 0;
X    contrl[3] = 2;
X    contrl[5] = 102;
X    contrl[6] = handle;
X    VDI_CALL();
X}
SHAR_EOF
chmod 0644 vdi5_102.c || echo "restore of vdi5_102.c fails"
echo "x - extracting vdi5_11.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_11.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
```

```
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_curaddress( WORD handle, WORD row, WORD column)
X{
X    intin[0] = row;
X    intin[1] = column;
X    contrl[0] = 5;
X    contrl[1] = 0;
X    contrl[3] = 2;
X    contrl[5] = 11;
X    contrl[6] = handle;
X    VDI_CALL();
X}
SHAR_EOF
chmod 0644 vdi5_11.c || echo "restore of vdi5_11.c fails"
echo "x - extracting vdi5_12.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_12.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_curtext( WORD handle, char *string)
```

```
X{
X    WORD tmp = 0;
X    while (intin[tmp++] = *string++);
X    contrl[0] = 5;
X    contrl[1] = 0;
X    contrl[3] = tmp - 1;    /* ohne nullbyte ? */
X    contrl[5] = 12;
X    contrl[6] = handle;
X    VDI_CALL();
X}
SHAR_EOF
chmod 0644 vdi5_12.c || echo "restore of vdi5_12.c fails"
echo "x - extracting vdi5_13.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_13.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_rvon( WORD handle)
X{
X    contrl[5] = 13;
X    vdi_macro_a(5, handle);
X}
X
X
SHAR_EOF
chmod 0644 vdi5_13.c || echo "restore of vdi5_13.c fails"
echo "x - extracting vdi5_14.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_14.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
```

```
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_rvoff( WORD handle)
X{
X    contrl[5] = 14;
X    vdi_macro_a(5, handle);
X}
X
X
SHAR_EOF
chmod 0644 vdi5_14.c || echo "restore of vdi5_14.c fails"
echo "x - extracting vdi5_15.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_15.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vq_curaddress( WORD handle, WORD *row, WORD *column)
X{
X    contrl[5] = 15;
X    vdi_macro_a(5, handle);
```

```
X    *row = intout[0];
X    *column = intout[1];
X}
X
X
SHAR_EOF
chmod 0644 vdi5_15.c || echo "restore of vdi5_15.c fails"
echo "x - extracting vdi5_16.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_16.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vq_tabstatus( WORD handle)
X{
X    contrl[5] = 16;
X    vdi_macro_a(5, handle);
X    return intout[0];
X}
X
X
SHAR_EOF
chmod 0644 vdi5_16.c || echo "restore of vdi5_16.c fails"
echo "x - extracting vdi5_17.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_17.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
```

```
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_hardcopy( WORD handle)
X{
X    contrl[5] = 17;
X    vdi_macro_a(5, handle);
X}
X
X
SHAR_EOF
chmod 0644 vdi5_17.c || echo "restore of vdi5_17.c fails"
echo "x - extracting vdi5_18.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_18.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_dspcur( WORD handle, WORD x, WORD y)
X{
X    ptsin[0] = x;
X    ptsin[1] = y;
X    contrl[0] = 5;
X    contrl[1] = 1;
X    contrl[3] = 0;
X    contrl[5] = 18;
```

```
X    contrl[6] = handle;
X    VDI_CALL();
X}
X
X
SHAR_EOF
chmod 0644 vdi5_18.c || echo "restore of vdi5_18.c fails"
echo "x - extracting vdi5_19.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_19.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_rmcur( WORD handle)
X{
X    contrl[5] = 19;
X    vdi_macro_a(5, handle);
X}
X
X
SHAR_EOF
chmod 0644 vdi5_19.c || echo "restore of vdi5_19.c fails"
echo "x - extracting vdi5_2.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_2.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
```

```
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_exit_cur( WORD handle)
X{
X    contrl[5] = 2;
X    vdi_macro_a(5, handle);
X}
X
SHAR_EOF
chmod 0644 vdi5_2.c || echo "restore of vdi5_2.c fails"
echo "x - extracting vdi5_20.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_20.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_form_adv( WORD handle)
X{
X    contrl[5] = 20;
X    vdi_macro_a(5, handle);
X}
X
X
SHAR_EOF
chmod 0644 vdi5_20.c || echo "restore of vdi5_20.c fails"
echo "x - extracting vdi5_21.c (Text)"
```

```
sed 's/^X//' << 'SHAR_EOF' > vdi5_21.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_output_window( WORD handle, WORD xyarray[])
X{
X    pblock[2] = xyarray;
X    contrl[0] = 5;
X    contrl[1] = 2;
X    contrl[3] = 0;
X    contrl[5] = 21;
X    contrl[6] = handle;
X    VDI_CALL();
X    pblock[2] = ptsin;
X}
SHAR_EOF
chmod 0644 vdi5_21.c || echo "restore of vdi5_21.c fails"
echo "x - extracting vdi5_22.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_22.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
```

```
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_clear_disp( WORD handle)
X{
X    contrl[5] = 22;
X    vdi_macro_a(5, handle);
X}
X
X
SHAR_EOF
chmod 0644 vdi5_22.c || echo "restore of vdi5_22.c fails"
echo "x - extracting vdi5_23.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_23.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_bit_image( WORD handle, char *filename, WORD aspect, WORD scaling, WORD
num_pts, WORD xyarray[])
X{
X    int i;
X    for(i=0; i<4; i++) ptsin[i] = xyarray[i];
X    intin[0] = aspect;
X    intin[1] = scaling;
X    i = 2;
X    while(intin[i++] = *filename++);
X    contrl[0] = 5;
X    contrl[1] = num_pts;
X    contrl[3] = --i;
X    contrl[5] = 23;
```

```
X    contrl[6] = handle;
X    VDI_CALL();
X}
SHAR_EOF
chmod 0644 vdi5_23.c || echo "restore of vdi5_23.c fails"
echo "x - extracting vdi5_3.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_3.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_enter_cur( WORD handle)
X{
X    contrl[5] = 3;
X    vdi_macro_a(5, handle);
X}
X
X
SHAR_EOF
chmod 0644 vdi5_3.c || echo "restore of vdi5_3.c fails"
echo "x - extracting vdi5_4.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_4.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
```

```
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_curup( WORD handle)
X{
X    contrl[5] = 4;
X    vdi_macro_a(5, handle);
X}
X
X
SHAR_EOF
chmod 0644 vdi5_4.c || echo "restore of vdi5_4.c fails"
echo "x - extracting vdi5_5.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_5.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_curdown( WORD handle)
X{
X    contrl[5] = 5;
X    vdi_macro_a(5, handle);
X}
X
X
SHAR_EOF
chmod 0644 vdi5_5.c || echo "restore of vdi5_5.c fails"
echo "x - extracting vdi5_6.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_6.c &&
```

```
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_curright( WORD handle)
X{
X    contrl[5] = 6;
X    vdi_macro_a(5, handle);
X}
X
X
SHAR_EOF
chmod 0644 vdi5_6.c || echo "restore of vdi5_6.c fails"
echo "x - extracting vdi5_7.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_7.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
```

```
X#include "gem-guts.h"
X
XVOID v_curleft( WORD handle)
X{
X    contrl[5] = 7;
X    vdi_macro_a(5, handle);
X}
X
X
SHAR_EOF
chmod 0644 vdi5_7.c || echo "restore of vdi5_7.c fails"
echo "x - extracting vdi5_8.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_8.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_curhome( WORD handle)
X{
X    contrl[5] = 8;
X    vdi_macro_a(5, handle);
X}
X
X
SHAR_EOF
chmod 0644 vdi5_8.c || echo "restore of vdi5_8.c fails"
echo "x - extracting vdi5_9.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_9.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
```

```
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_eeos( WORD handle)
X{
X    contrl[5] = 9;
X    vdi_macro_a(5, handle);
X}
X
X
SHAR_EOF
chmod 0644 vdi5_9.c || echo "restore of vdi5_9.c fails"
echo "x - extracting vdi5_98.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_98.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID    v_meta_extents( WORD handle, WORD min_x, WORD min_y, WORD max_x, WORD
max_y)
X{
X    ptsin[0] = min_x;
X    ptsin[1] = min_y;
```

```
X    ptsin[2] = max_x;
X    ptsin[3] = max_y;
X    contrl[0] = 5;
X    contrl[1] = 2;
X    contrl[3] = 0;
X    contrl[5] = 98;
X    contrl[6] = handle;
X    VDI_CALL();
X}
SHAR_EOF
chmod 0644 vdi5_98.c || echo "restore of vdi5_98.c fails"
echo "x - extracting vdi5_99.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi5_99.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_write_meta( WORD handle, WORD num_intin, WORD a_intin[], WORD num_ptsin,
WORD a_ptsin[])
X{
X    pblock[1] = a_intin;
X    pblock[2] = a_ptsin;
X    contrl[0] = 5;
X    contrl[1] = num_ptsin;
X    contrl[3] = num_intin;
X    contrl[5] = 99;
X    contrl[6] = handle;
X    VDI_CALL();
X
X}
SHAR_EOF
chmod 0644 vdi5_99.c || echo "restore of vdi5_99.c fails"
echo "x - extracting vdi_1.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_1.c &&
X/*
```

```
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
Xvoid v_opnwk(WORD work_in[], WORD * handle, WORD work_out[])
X{
X    pblock[0] = contrl;
X    pblock[1] = work_in;
X    pblock[2] = ptsin;
X    pblock[3] = work_out;
X    pblock[4] = work_out+45;
X
X    contrl[0] = 1;
X    contrl[1] = 0;
X    contrl[3] = 11;
X
X    VDI_CALL();
X
X    *handle = contrl[6];
X    pblock[1] = intin;      /* standardbelegung wieder herstellen */
X    pblock[3] = intout;
X    pblock[4] = ptsout;
X}
X
SHAR_EOF
chmod 0644 vdi_1.c || echo "restore of vdi_1.c fails"
echo "x - extracting vdi_10.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_10.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
```

```
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_cellarray(WORD handle, WORD pxyarray[], WORD row_length, WORD el_used,
WORD num_rows, WORD wrt_mode, WORD colarray[])
X{
X    pblock[1] = colarray;
X    pblock[2] = pxyarray;
X    contrl[0] = 10;
X    contrl[1] = 2;
X    contrl[3] = row_length * num_rows;
X    contrl[6] = handle;
X    contrl[7] = row_length;
X    contrl[8] = el_used;
X    contrl[9] = num_rows;
X    contrl[10]= wrt_mode;
X    VDI_CALL();
X    pblock[1] = intin;
X    pblock[2] = intout;
X}
SHAR_EOF
chmod 0644 vdi_10.c || echo "restore of vdi_10.c fails"
echo "x - extracting vdi_100.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_100.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
```

```
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
Xvoid v_opnvwk(WORD work_in[], WORD * handle, WORD work_out[])
X{
X    pblock[0] = contrl;
X    pblock[1] = work_in;
X    pblock[2] = ptsin;
X    pblock[3] = work_out;
X    pblock[4] = work_out+45;
X
X    contrl[0] = 100;
X    contrl[1] = 0;
X    contrl[3] = 11;
X    contrl[6] = *handle;
X
X    VDI_CALL();
X
X    *handle   = contrl[6];
X    pblock[1] = intin;     /* standardbelegung wieder herstellen */
X    pblock[3] = intout;
X    pblock[4] = ptsout;
X}
X
SHAR_EOF
chmod 0644 vdi_100.c || echo "restore of vdi_100.c fails"
echo "x - extracting vdi_101.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_101.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_clsvwk(WORD handle) { vdi_macro_a(101, handle); }
SHAR_EOF
```

```
chmod 0644 vdi_101.c || echo "restore of vdi_101.c fails"
echo "x - extracting vdi_102.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_102.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vq_extnd(WORD handle, WORD owflag, WORD work_out[])
X{
X    pblock[3] = work_out;
X    pblock[4] = work_out+45;
X    intin[0] = owflag;
X    contrl[0] = 102;
X    contrl[1] = 0;
X    contrl[3] = 1;
X    contrl[6] = handle;
X    VDI_CALL();
X    pblock[3] = intout;
X    pblock[4] = ptsout;
X}
SHAR_EOF
chmod 0644 vdi_102.c || echo "restore of vdi_102.c fails"
echo "x - extracting vdi_103.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_103.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
```

```
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_contourfill(WORD handle, WORD x, WORD y, WORD index)
X{
X    intin[0] = index;
X    ptsin[0] = x;
X    ptsin[1] = y;
X    contrl[0] = 103;
X    contrl[1] = 1;
X    contrl[3] = 1;
X    contrl[6] = handle;
X    VDI_CALL();
X}
X
SHAR_EOF
chmod 0644 vdi_103.c || echo "restore of vdi_103.c fails"
echo "x - extracting vdi_104.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_104.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vsf_perimeter(WORD handle, WORD per_vis)
X{ return vdi_macro_b(104, handle, per_vis); }
X
SHAR_EOF
```

```
chmod 0644 vdi_104.c || echo "restore of vdi_104.c fails"
echo "x - extracting vdi_105.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_105.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_get_pixel(WORD handle, WORD x, WORD y, WORD *pel, WORD *index)
X{
X    ptsin[0] = x;
X    ptsin[1] = y;
X    contrl[0] = 105;
X    contrl[1] = 1;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
X    *pel = intout[0];
X    *index = intout[1];
X}
SHAR_EOF
chmod 0644 vdi_105.c || echo "restore of vdi_105.c fails"
echo "x - extracting vdi_106.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_106.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
```

```
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vst_effects(WORD handle, WORD effect)
X{ return vdi_macro_b(106, handle, effect); }
SHAR_EOF
chmod 0644 vdi_106.c || echo "restore of vdi_106.c fails"
echo "x - extracting vdi_107.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_107.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vst_point(WORD handle, WORD point, WORD *char_width, WORD *char_height, WORD
*cell_width, WORD *cell_height)
X{
X    intin[0] = point;
X    contrl[0] = 107;
X    contrl[1] = 0;
X    contrl[3] = 1;
X    contrl[6] = handle;
X    VDI_CALL();
X    *char_width = ptsout[0];
X    *char_height = ptsout[1];
X    *cell_width = ptsout[2];
X    *cell_height = ptsout[3];
X    return intout[0];
X}
```

```
SHAR_EOF
chmod 0644 vdi_107.c || echo "restore of vdi_107.c fails"
echo "x - extracting vdi_108.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_108.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vsl_ends(WORD handle, WORD beg_style, WORD end_style)
X{
X    intin[0] = beg_style;
X    intin[1] = end_style;
X    contrl[0] = 108;
X    contrl[1] = 0;
X    contrl[3] = 2;
X    contrl[6] = handle;
X    VDI_CALL();
X}
SHAR_EOF
chmod 0644 vdi_108.c || echo "restore of vdi_108.c fails"
echo "x - extracting vdi_109.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_109.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
```

```
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vro_cpyfm(WORD handle, WORD wr_mode, WORD pxyarray[], FDB *psrcMFDB, FDB
*pdesMFDB)
X{
X    LONG *tmp;
X    intin[0]  = wr_mode;
X    pblock[2] = pxyarray;
X    tmp = (LONG *) &contrl[7];
X    *tmp++ = (LONG) psrcMFDB;
X    *tmp   = (LONG) pdesMFDB;
X    contrl[0] = 109;
X    contrl[1] = 4;
X    contrl[3] = 1;
X    contrl[6] = handle;
X    VDI_CALL();
X    pblock[2] = ptsin;
X}
SHAR_EOF
chmod 0644 vdi_109.c || echo "restore of vdi_109.c fails"
echo "x - extracting vdi_110.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_110.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
SHAR_EOF
echo "End of part 4"
echo "File vdi_110.c is continued in part 5"
```

```
echo "5" > s2_seq_.tmp
exit 0


From -7388814550491506403
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,17ad2b1af752d155,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-20 16:30:42 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!ames!apple!sun-barr!newstop!sun!
grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i060: gccgem -- GEM libraries for GCC  part05/06
Keywords: shar, C
Message-ID: <33908@grapevine.uucp>
Date: 20 Jul 89 23:30:42 GMT
Sender: news@grapevine.uucp
Lines: 1332
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: uunet!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 2, Issue 60
Archive-name: gccgem/part05

#!/bin/sh
# this is part 5 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file vdi_110.c continued
#
CurArch=5
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file vdi_110.c"
sed 's/^X//' << 'SHAR_EOF' >> vdi_110.c
X
XVOID vr_trnfm(WORD handle, FDB *psrcMFDB, FDB *pdesMFDB)
X{
X    LONG *tmp = (LONG *) &contrl[7];
X    *tmp++ = (LONG) psrcMFDB;
X    *tmp   = (LONG) pdesMFDB;
X    contrl[0] = 110;
X    contrl[1] = 0;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
```

```
X}
SHAR_EOF
echo "File vdi_110.c is complete"
chmod 0644 vdi_110.c || echo "restore of vdi_110.c fails"
echo "x - extracting vdi_111.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_111.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vsc_form(WORD handle, WORD pcur_form[])
X{
X    pblock[1] = pcur_form;
X    contrl[0] = 111;
X    contrl[1] = 0;
X    contrl[3] = 37;
X    contrl[6] = handle;
X    VDI_CALL();
X    pblock[1] = intin;
X}
SHAR_EOF
chmod 0644 vdi_111.c || echo "restore of vdi_111.c fails"
echo "x - extracting vdi_112.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_112.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
```

```
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vsf_updat(WORD handle, WORD pfill_pat[], WORD planes)
X{
X    pblock[1] = pfill_pat;
X    contrl[0] = 112;
X    contrl[1] = 0;
X    contrl[3] = planes<<4;  /* mal 16 */
X    contrl[6] = handle;
X    VDI_CALL();
X    pblock[1] = intin;
X}
SHAR_EOF
chmod 0644 vdi_112.c || echo "restore of vdi_112.c fails"
echo "x - extracting vdi_113.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_113.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vsl_udsty(WORD handle, WORD pattern)
X{   vdi_macro_d(113, handle, pattern); }
SHAR_EOF
chmod 0644 vdi_113.c || echo "restore of vdi_113.c fails"
echo "x - extracting vdi_114.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_114.c &&
```

```
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vr_recfl(WORD handle, WORD pxyarray[])
X{
X    pblock[2] = pxyarray;
X    contrl[0] = 114;
X    contrl[1] = 2;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
X    pblock[2] = ptsin;
X}
SHAR_EOF
chmod 0644 vdi_114.c || echo "restore of vdi_114.c fails"
echo "x - extracting vdi_115.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_115.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
```

```
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID    vqin_mode( WORD handle, WORD dev_type, WORD *input_mode)
X{
X    vdi_macro_d(115, handle, dev_type);
X    *input_mode = intout[0];
X}
SHAR_EOF
chmod 0644 vdi_115.c || echo "restore of vdi_115.c fails"
echo "x - extracting vdi_116.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_116.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vqt_extent(WORD handle, char *string, WORD extent[])
X{
X    int i = 0;
X    while (intin[i] = *string++) i++;
X    pblock[4] = extent;
X    contrl[0] = 116;
X    contrl[1] = 0;
X    contrl[3] = i;
X    contrl[6] = handle;
X    VDI_CALL();
X    pblock[4] = ptsout;
X}
SHAR_EOF
chmod 0644 vdi_116.c || echo "restore of vdi_116.c fails"
echo "x - extracting vdi_117.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_117.c &&
X/*
```

```
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vqt_width(WORD handle, WORD character, WORD *cell_width, WORD *left_delta,
WORD *right_delta)
X{
X    intin[0] = character;
X    contrl[0] = 117;
X    contrl[0] = 0;
X    contrl[0] = 1;
X    contrl[0] = handle;
X    VDICALL();
X    *cell_width = ptsout[0];
X    *left_delta = ptsout[2];
X    *right_delta = ptsout[4];
X    return intout[0];
X}
SHAR_EOF
chmod 0644 vdi_117.c || echo "restore of vdi_117.c fails"
echo "x - extracting vdi_118.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_118.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
```

```
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vex_timv(WORD handle, LONG tim_addr, LONG *otim_addr, WORD *tim_conv)
X{
X    *((LONG *) &contrl[7]) = tim_addr;
X    contrl[0] = 118;
X    contrl[1] = 0;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
X    *otim_addr = *((LONG *) &contrl[9]);
X    *tim_conv = intout[0];
X}
SHAR_EOF
chmod 0644 vdi_118.c || echo "restore of vdi_118.c fails"
echo "x - extracting vdi_119.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_119.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vst_load_fonts(WORD handle, WORD select)
X{ return vdi_macro_b(119, handle, select); }
SHAR_EOF
chmod 0644 vdi_119.c || echo "restore of vdi_119.c fails"
echo "x - extracting vdi_11_1.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_11_1.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
```

```
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_bar(WORD handle, WORD pxyarray[])
X{
X    pblock[2] = pxyarray;
X    contrl[0] = 11;
X    contrl[1] = 2;
X    contrl[3] = 0;
X    contrl[5] = 1;
X    contrl[6] = handle;
X    VDI_CALL();
X    pblock[2] = ptsin;
X}
SHAR_EOF
chmod 0644 vdi_11_1.c || echo "restore of vdi_11_1.c fails"
echo "x - extracting vdi_11_2.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_11_2.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
```

```
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_arc(WORD handle, WORD x, WORD y, WORD radius, WORD begang, WORD endang)
X{
X    ptsin[0] = x;
X    ptsin[1] = y;
X    ptsin[2] = 0;
X    ptsin[3] = 0;
X    ptsin[4] = 0;
X    ptsin[5] = 0;
X    ptsin[6] = radius;
X    ptsin[7] = 0;
X    intin[0] = begang;
X    intin[1] = endang;
X    contrl[0] = 11;
X    contrl[1] = 4;
X    contrl[3] = 2;
X    contrl[5] = 2;
X    contrl[6] = handle;
X    VDI_CALL();
X}
SHAR_EOF
chmod 0644 vdi_11_2.c || echo "restore of vdi_11_2.c fails"
echo "x - extracting vdi_11_3.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_11_3.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_pie(WORD handle, WORD x, WORD y, WORD radius, WORD begang, WORD endang)
X{
X    ptsin[0] = x;
X    ptsin[1] = y;
```

```
X    ptsin[2] = 0;
X    ptsin[3] = 0;
X    ptsin[4] = 0;
X    ptsin[5] = 0;
X    ptsin[6] = radius;
X    ptsin[7] = 0;
X    intin[0] = begang;
X    intin[1] = endang;
X    contrl[0] = 11;
X    contrl[1] = 4;
X    contrl[3] = 2;
X    contrl[5] = 3;
X    contrl[6] = handle;
X    VDI_CALL();
X}
SHAR_EOF
chmod 0644 vdi_11_3.c || echo "restore of vdi_11_3.c fails"
echo "x - extracting vdi_11_4.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_11_4.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_circle(WORD handle, WORD x, WORD y, WORD radius)
X{
X    ptsin[0] = x;
X    ptsin[1] = y;
X    ptsin[2] = 0;
X    ptsin[3] = 0;
X    ptsin[4] = radius;
X    ptsin[5] = 0;
X    contrl[0] = 11;
X    contrl[1] = 3;
X    contrl[3] = 0;
X    contrl[5] = 4;
```

```
X    contrl[6] = handle;
X    VDI_CALL();
X}
SHAR_EOF
chmod 0644 vdi_11_4.c || echo "restore of vdi_11_4.c fails"
echo "x - extracting vdi_11_5.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_11_5.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_ellipse(WORD handle, WORD x, WORD y, WORD xradius, WORD yradius)
X{
X    ptsin[0] = x;
X    ptsin[1] = y;
X    ptsin[2] = xradius;
X    ptsin[3] = yradius;
X    contrl[0] = 11;
X    contrl[1] = 2;
X    contrl[3] = 0;
X    contrl[5] = 5;
X    contrl[6] = handle;
X    VDI_CALL();
X}
SHAR_EOF
chmod 0644 vdi_11_5.c || echo "restore of vdi_11_5.c fails"
echo "x - extracting vdi_11_6.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_11_6.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
```

```
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_ellarc(WORD handle, WORD x, WORD y, WORD xradius, WORD yradius, WORD
begang, WORD endang)
X{
X    ptsin[0] = x;
X    ptsin[1] = y;
X    ptsin[2] = xradius;
X    ptsin[3] = yradius;
X    intin[0] = begang;
X    intin[1] = endang;
X    contrl[0] = 11;
X    contrl[1] = 2;
X    contrl[3] = 2;
X    contrl[5] = 6;
X    contrl[6] = handle;
X    VDI_CALL();
X}
SHAR_EOF
chmod 0644 vdi_11_6.c || echo "restore of vdi_11_6.c fails"
echo "x - extracting vdi_11_7.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_11_7.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
```

```
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_ellpie(WORD handle, WORD x, WORD y, WORD xradius, WORD yradius, WORD
begang, WORD endang)
X{
X    ptsin[0] = x;
X    ptsin[1] = y;
X    ptsin[2] = xradius;
X    ptsin[3] = yradius;
X    intin[0] = begang;
X    intin[1] = endang;
X    contrl[0] = 11;
X    contrl[1] = 2;
X    contrl[3] = 2;
X    contrl[5] = 7;
X    contrl[6] = handle;
X    VDI_CALL();
X}
SHAR_EOF
chmod 0644 vdi_11_7.c || echo "restore of vdi_11_7.c fails"
echo "x - extracting vdi_11_8.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_11_8.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_rbox(WORD handle, WORD xyarray[])
X{
X    pblock[2] = xyarray;
X    contrl[0] = 11;
X    contrl[1] = 2;
X    contrl[3] = 0;
X    contrl[5] = 8;
```

```
X    contrl[6] = handle;
X    VDI_CALL();
X    pblock[2] = ptsin;
X}
SHAR_EOF
chmod 0644 vdi_11_8.c || echo "restore of vdi_11_8.c fails"
echo "x - extracting vdi_11_9.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_11_9.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_rfbox(WORD handle, WORD xyarray[])
X{
X    pblock[2] = xyarray;
X    contrl[0] = 11;
X    contrl[1] = 2;
X    contrl[3] = 0;
X    contrl[5] = 9;
X    contrl[6] = handle;
X    VDI_CALL();
X    pblock[2] = ptsin;
X}
SHAR_EOF
chmod 0644 vdi_11_9.c || echo "restore of vdi_11_9.c fails"
echo "x - extracting vdi_12.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_12.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
```

```
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vst_height(WORD handle, WORD height, WORD *char_width, WORD *char_height,
WORD *cell_width, WORD *cell_height)
X{
X    ptsin[0] = 0;
X    ptsin[1] = height;
X    contrl[0] = 12;
X    contrl[1] = 1;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
X    *char_width = ptsout[0];
X    *char_height = ptsout[1];
X    *cell_width = ptsout[2];
X    *cell_height = ptsout[3];
X}
X
SHAR_EOF
chmod 0644 vdi_12.c || echo "restore of vdi_12.c fails"
echo "x - extracting vdi_120.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_120.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
```

```
X */
X#include "gem-guts.h"
X
XVOID vst_unload_fonts(WORD handle, WORD select)
X{   vdi_macro_d(120, handle, select); }
SHAR_EOF
chmod 0644 vdi_120.c || echo "restore of vdi_120.c fails"
echo "x - extracting vdi_121.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_121.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vrt_cpyfm(WORD handle, WORD wr_mode, WORD pxyarray[], FDB *psrcMFDB, FDB
*pdesMFDB, WORD color_index[])
X{
X    LONG *tmp = (LONG *) &contrl[7];
X    intin[0]  = wr_mode;
X    intin[1]  = color_index[0];
X    intin[2]  = color_index[1];
X    pblock[2] = pxyarray;
X    *tmp++ = (LONG) psrcMFDB;
X    *tmp   = (LONG) pdesMFDB;
X    contrl[0] = 121;
X    contrl[1] = 4;
X    contrl[3] = 3;
X    contrl[6] = handle;
X    VDI_CALL();
X    pblock[2] = ptsin;
X}
SHAR_EOF
chmod 0644 vdi_121.c || echo "restore of vdi_121.c fails"
echo "x - extracting vdi_122.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_122.c &&
X/*
```

```
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_show_c(WORD handle, WORD reset)
X{
X    intin[0] = reset;
X    contrl[0] = 122;
X    contrl[1] = 0;
X    contrl[3] = 1;
X    contrl[6] = handle;
X    VDI_CALL();
X}
X
SHAR_EOF
chmod 0644 vdi_122.c || echo "restore of vdi_122.c fails"
echo "x - extracting vdi_123.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_123.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
```

```
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID  v_hide_c(WORD handle)
X{ vdi_macro_a(123, handle); }
X
SHAR_EOF
chmod 0644 vdi_123.c || echo "restore of vdi_123.c fails"
echo "x - extracting vdi_124.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_124.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vq_mouse(WORD handle, WORD *pstatus, WORD *px, WORD *py)
X{
X    contrl[0] = 124;
X    contrl[1] = 0;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
X    *pstatus = intout[0];
X    *px = ptsout[0];
X    *py = ptsout[1];
X}
SHAR_EOF
chmod 0644 vdi_124.c || echo "restore of vdi_124.c fails"
echo "x - extracting vdi_125.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_125.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
```

```
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vex_butv(WORD handle, LONG pusrcode, LONG *psavcode)
X{ vdi_macro_e(125, handle, pusrcode, psavcode); }
SHAR_EOF
chmod 0644 vdi_125.c || echo "restore of vdi_125.c fails"
echo "x - extracting vdi_126.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_126.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vex_motv(WORD handle, LONG pusrcode, LONG *psavcode)
X{ vdi_macro_e(126, handle, pusrcode, psavcode); }
X
SHAR_EOF
chmod 0644 vdi_126.c || echo "restore of vdi_126.c fails"
echo "x - extracting vdi_127.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_127.c &&
X/*
```

```
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vex_curv(WORD handle, LONG pusrcode, LONG *psavcode)
X{ vdi_macro_e(127, handle, pusrcode, psavcode); }
X
SHAR_EOF
chmod 0644 vdi_127.c || echo "restore of vdi_127.c fails"
echo "x - extracting vdi_128.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_128.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vq_key_s(WORD handle, WORD *pstatus)
X{
X    contrl[0] = 128;
```

```
X    contrl[1] = 0;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
X    *pstatus = intout[0];
X}
SHAR_EOF
chmod 0644 vdi_128.c || echo "restore of vdi_128.c fails"
echo "x - extracting vdi_129.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_129.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vs_clip(WORD handle, WORD clip_flag, WORD pxyarray[])
X{
X    pblock[2] = pxyarray;
X    intin[0]  = clip_flag;
X    contrl[0] = 129;
X    contrl[1] = 2;
X    contrl[3] = 1;
X    contrl[6] = handle;
X    VDI_CALL();
X    pblock[2] = ptsin;
X}
SHAR_EOF
chmod 0644 vdi_129.c || echo "restore of vdi_129.c fails"
echo "x - extracting vdi_13.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_13.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
```

```
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vst_rotation(WORD handle, WORD angle)
X{ return vdi_macro_b(13, handle, angle); }
SHAR_EOF
chmod 0644 vdi_13.c || echo "restore of vdi_13.c fails"
echo "x - extracting vdi_130.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_130.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD    vqt_name( WORD handle, WORD element_num, char name[])
X{
X    int tmp;
X    contrl[0] = 130;
X    contrl[1] = 0;
X    contrl[3] = 1;
X    contrl[6] = handle;
X    VDI_CALL();
X    for(tmp=0; tmp<32; tmp++)
```

```
X        name[tmp]=intout[tmp+1];
X    return intout[0];
X}
SHAR_EOF
chmod 0644 vdi_130.c || echo "restore of vdi_130.c fails"
echo "x - extracting vdi_131.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_131.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID    vqt_fontinfo( WORD handle, WORD *minADE, WORD *maxADE, WORD distances[],
XWORD *maxwidth, WORD effects[])
X{
X    vdi_macro_a(131, handle);
X    *minADE = intout[0];
X    *maxADE = intout[1];
X    *maxwidth = ptsout[0];
X    distances[0] = ptsout[1];
X    distances[1] = ptsout[3];
X    distances[2] = ptsout[5];
X    distances[3] = ptsout[7];
X    distances[4] = ptsout[9];
X    effects[0] = ptsout[2];
X    effects[1] = ptsout[4];
X    effects[2] = ptsout[6];
X}
SHAR_EOF
chmod 0644 vdi_131.c || echo "restore of vdi_131.c fails"
echo "x - extracting vdi_14.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_14.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
```

```
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vs_color(WORD handle, WORD index, WORD rgb_in[])
X{
X    contrl[0] = 14;
X    contrl[1] = 0;
X    contrl[3] = 4;
X    contrl[6] = handle;
X    intin[0]  = index;
X    intin[1]  = rgb_in[0];
X    intin[2]  = rgb_in[1];
X    intin[3]  = rgb_in[2];
X    intin[4]  = rgb_in[3];
X    VDI_CALL();
X}
X
SHAR_EOF
chmod 0644 vdi_14.c || echo "restore of vdi_14.c fails"
echo "x - extracting vdi_15.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_15.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
```

```
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vsl_type(WORD handle, WORD style)
X{ return vdi_macro_b(15, handle, style); }
X
SHAR_EOF
chmod 0644 vdi_15.c || echo "restore of vdi_15.c fails"
echo "x - extracting vdi_16.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_16.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vsl_width(WORD handle, WORD width)
X{
X    ptsin[0] = width;
X    ptsin[1] = 0;
X    contrl[0] = 16;
X    contrl[1] = 1;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
X    return ptsout[0];
X}
X
SHAR_EOF
chmod 0644 vdi_16.c || echo "restore of vdi_16.c fails"
echo "x - extracting vdi_17.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_17.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
```

```
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vsl_color(WORD handle, WORD color_index)
X{ return vdi_macro_b(17, handle, color_index); }
X
SHAR_EOF
chmod 0644 vdi_17.c || echo "restore of vdi_17.c fails"
echo "x - extracting vdi_18.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_18.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
SHAR_EOF
echo "End of part 5"
echo "File vdi_18.c is continued in part 6"
echo "6" > s2_seq_.tmp
exit 0
```


```
From -1391116969028797136
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,e09a09684f919907,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-20 16:30:55 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!ames!apple!sun-barr!newstop!sun!
grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i061: gccgem -- GEM libraries for GCC  part06/06
Keywords: shar, C
Message-ID: <33909@grapevine.uucp>
Date: 20 Jul 89 23:30:55 GMT
Sender: news@grapevine.uucp
Lines: 1162
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: uunet!unido!sbsvax!roeder (Edgar Roeder)
```

```
Posting-number: Volume 2, Issue 61
Archive-name: gccgem/part06

#!/bin/sh
# this is part 6 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file vdi_18.c continued
#
CurArch=6
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
echo "x - Continuing file vdi_18.c"
sed 's/^X//' << 'SHAR_EOF' >> vdi_18.c
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vsm_type(WORD handle, WORD symbol)
X{ return vdi_macro_b(18, handle, symbol); }
SHAR_EOF
echo "File vdi_18.c is complete"
chmod 0644 vdi_18.c || echo "restore of vdi_18.c fails"
echo "x - extracting vdi_19.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_19.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
```

```
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vsm_height(WORD handle, WORD height)
X{
X    ptsin[0] = 0;
X    ptsin[1] = height;
X    contrl[0] = 19;
X    contrl[1] = 1;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
X    return ptsout[1];
X}
X
SHAR_EOF
chmod 0644 vdi_19.c || echo "restore of vdi_19.c fails"
echo "x - extracting vdi_2.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_2.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
```

```
X
XVOID v_clswk(WORD handle) { vdi_macro_a(2, handle); }
SHAR_EOF
chmod 0644 vdi_2.c || echo "restore of vdi_2.c fails"
echo "x - extracting vdi_20.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_20.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vsm_color(WORD handle, WORD color_index)
X{   return vdi_macro_b(20, handle, color_index); }
X
SHAR_EOF
chmod 0644 vdi_20.c || echo "restore of vdi_20.c fails"
echo "x - extracting vdi_21.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_21.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
```

```
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vst_font(WORD handle, WORD font)
X{ return vdi_macro_b(21, handle, font); }
SHAR_EOF
chmod 0644 vdi_21.c || echo "restore of vdi_21.c fails"
echo "x - extracting vdi_22.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_22.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vst_color(WORD handle, WORD color_index)
X{ return vdi_macro_b(22, handle, color_index); }
SHAR_EOF
chmod 0644 vdi_22.c || echo "restore of vdi_22.c fails"
echo "x - extracting vdi_23.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_23.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
```

```
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vsf_interior(WORD handle, WORD style)
X{ return vdi_macro_b(23, handle, style); }
SHAR_EOF
chmod 0644 vdi_23.c || echo "restore of vdi_23.c fails"
echo "x - extracting vdi_24.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_24.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vsf_style(WORD handle, WORD style_index)
X{ return vdi_macro_b(24, handle, style_index); }
SHAR_EOF
chmod 0644 vdi_24.c || echo "restore of vdi_24.c fails"
echo "x - extracting vdi_25.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_25.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
```

```
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vsf_color(WORD handle, WORD color_index)
X{ return vdi_macro_b(25, handle, color_index); }
SHAR_EOF
chmod 0644 vdi_25.c || echo "restore of vdi_25.c fails"
echo "x - extracting vdi_26.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_26.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vq_color(WORD handle, WORD color_index, WORD set_flag, WORD *rgb)
X{
X    intin[0] = color_index;
X    intin[1] = set_flag;
X    contrl[0] = 26;
X    contrl[1] = 0;
X    contrl[3] = 2;
X    contrl[6] = handle;
X    VDI_CALL();
X    *rgb++ = intout[1];
X    *rgb++ = intout[2];
X    *rgb   = intout[3];
X}
SHAR_EOF
chmod 0644 vdi_26.c || echo "restore of vdi_26.c fails"
```

```
echo "x - extracting vdi_27.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_27.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID    vq_cellarray( WORD handle, WORD pxyarray[], WORD row_length, WORD
num_rows, WORD *el_used, WORD *rows_used, WORD *status, WORD colarray[])
X{
X    pblock[2] = pxyarray;
X    pblock[3] = colarray;
X    contrl[0] = 27;
X    contrl[1] = 2;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    contrl[7] = row_length;
X    contrl[8] = num_rows;
X    VDI_CALL();
X    *el_used = contrl[9];
X    *rows_used = contrl[10];
X    *status = contrl[11];
X    pblock[2] = ptsin;
X    pblock[3] = intout;
X}
X
SHAR_EOF
chmod 0644 vdi_27.c || echo "restore of vdi_27.c fails"
echo "x - extracting vdi_28.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_28.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
```

```
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vrq_locator(WORD handle, WORD x, WORD y, WORD *xout, WORD *yout, WORD *term)
X{
X    ptsin[0] = x;
X    ptsin[0] = x;
X    contrl[0] = 28;
X    contrl[1] = 1;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
X    *xout = ptsout[0];
X    *yout = ptsout[1];
X    *term = intout[0];
X}
X
XWORD vsm_locator(WORD handle, WORD x, WORD y, WORD *xout, WORD *yout, WORD *term)
X{
X    vrq_locator(handle, x, y, xout, yout, term);
X    return (contrl[4]<<1|contrl[2]);
X}
SHAR_EOF
chmod 0644 vdi_28.c || echo "restore of vdi_28.c fails"
echo "x - extracting vdi_29.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_29.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
```

```
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vrq_valuator(WORD handle, WORD valuator_in, WORD *valuator_out, WORD
*terminator)
X{
X    intin[0] = valuator_in;
X    contrl[0] = 29;
X    contrl[1] = 0;
X    contrl[3] = 1;
X    contrl[6] = handle;
X    VDI_CALL();
X    *valuator_out = intout[0];
X    *terminator   = intout[1];
X}
X
XVOID vsm_valuator(WORD handle, WORD val_in, WORD *val_out, WORD *term, WORD
*status)
X{
X    vrq_valuator(handle, val_in, val_out, term);
X    *status = contrl[4];
X}
SHAR_EOF
chmod 0644 vdi_29.c || echo "restore of vdi_29.c fails"
echo "x - extracting vdi_3.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_3.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
```

```
XVOID v_clrwk(WORD handle) {     vdi_macro_a(3, handle); }
SHAR_EOF
chmod 0644 vdi_3.c || echo "restore of vdi_3.c fails"
echo "x - extracting vdi_30.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_30.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vrq_choice(WORD handle, WORD ch_in, WORD *ch_out)
X{
X    intin[0] = ch_in;
X    contrl[0] = 30;
X    contrl[1] = 0;
X    contrl[3] = 1;
X    contrl[6] = handle;
X    VDI_CALL();
X    *ch_out = intout[0];
X}
X
XWORD vsm_choice(WORD handle, WORD *choice)
X{
X    contrl[0] = 30;
X    contrl[1] = 0;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
X    *choice = intout[0];
X    return contrl[4];
X}
SHAR_EOF
chmod 0644 vdi_30.c || echo "restore of vdi_30.c fails"
echo "x - extracting vdi_31.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_31.c &&
X/*
```

```
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vrq_string(WORD handle, WORD max_length, WORD echo_mode, WORD echo_xy[],
char *string)
X{
X    int i;
X    intin[0] = max_length;
X    intin[1] = echo_mode;
X    pblock[2] = echo_xy;
X    contrl[0] = 31;
X    contrl[1] = echo_mode;
X    contrl[3] = 2;
X    contrl[6] = handle;
X    VDI_CALL();
X    for (i=0; i<contrl[4];i++)
X         *string++ = intout[i];
X    *string = '\0';
X    pblock[2] = ptsin;
X}
X
XWORD vsm_string(WORD handle, WORD max_length, WORD echo_mode, WORD echo_xy[],
char *string)
X{
X    vrq_string(handle, max_length, echo_mode, echo_xy, string);
X    return contrl[4];
X}
X
SHAR_EOF
chmod 0644 vdi_31.c || echo "restore of vdi_31.c fails"
echo "x - extracting vdi_32.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_32.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
```

```
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vswr_mode(WORD handle, WORD mode)
X{ return vdi_macro_b(32, handle, mode); }
X
SHAR_EOF
chmod 0644 vdi_32.c || echo "restore of vdi_32.c fails"
echo "x - extracting vdi_33.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_33.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vsin_mode(WORD handle, WORD dev_type, WORD mode)
X{
X    intin[0] = dev_type;
X    intin[1] = mode;
X    contrl[0]= 33;
```

```
X    contrl[2]= 0;
X    contrl[3]= 2;
X    contrl[6]= handle;
X    VDI_CALL();
X    return intout[0];
X}
SHAR_EOF
chmod 0644 vdi_33.c || echo "restore of vdi_33.c fails"
echo "x - extracting vdi_35.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_35.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vql_attributes(WORD handle, WORD attrib[])
X{
X    pblock[3] = attrib;
X    contrl[0] = 35;
X    contrl[1] = 0;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
X    pblock[3] = intout;
X    attrib[5] = ptsout[0];
X}
SHAR_EOF
chmod 0644 vdi_35.c || echo "restore of vdi_35.c fails"
echo "x - extracting vdi_36.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_36.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
```

```
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vqm_attributes(WORD handle, WORD attrib[])
X{
X    pblock[3] = attrib;
X    contrl[0] = 36;
X    contrl[1] = 0;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
X    pblock[3] = intout;
X    attrib[3] = ptsout[0];
X    attrib[4] = ptsout[1];
X}
SHAR_EOF
chmod 0644 vdi_36.c || echo "restore of vdi_36.c fails"
echo "x - extracting vdi_37.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_37.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
```

```
X
XVOID vqf_attributes(WORD handle, WORD attrib[])
X{
X    pblock[3] = attrib;
X    contrl[0] = 37;
X    contrl[1] = 0;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
X    pblock[3] = intout;
X}
SHAR_EOF
chmod 0644 vdi_37.c || echo "restore of vdi_37.c fails"
echo "x - extracting vdi_38.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_38.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vqt_attributes(WORD handle, WORD attrib[])
X{
X    pblock[3] = attrib;
X    pblock[4] = attrib+6;
X    contrl[0] = 38;
X    contrl[1] = 0;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
X    pblock[3] = intout;
X    pblock[4] = ptsout;
X}
SHAR_EOF
chmod 0644 vdi_38.c || echo "restore of vdi_38.c fails"
echo "x - extracting vdi_39.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_39.c &&
```

```
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vst_alignment(WORD handle, WORD hor_in, WORD vert_in, WORD *hor_out, WORD
*vert_out)
X{
X    intin[0]  = hor_in;
X    intin[1]  = vert_in;
X    contrl[0] = 39;
X    contrl[1] = 0;
X    contrl[3] = 2;
X    contrl[6] = handle;
X    VDI_CALL();
X    *hor_out  = intout[0];
X    *vert_out = intout[1];
X}
SHAR_EOF
chmod 0644 vdi_39.c || echo "restore of vdi_39.c fails"
echo "x - extracting vdi_4.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_4.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
```

```
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_updwk(WORD handle) {     vdi_macro_a(4, handle); }
SHAR_EOF
chmod 0644 vdi_4.c || echo "restore of vdi_4.c fails"
echo "x - extracting vdi_6.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_6.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_pline(WORD handle, WORD count, WORD pxyarray[])
X{  vdi_macro_c(6, handle, count, pxyarray); }
SHAR_EOF
chmod 0644 vdi_6.c || echo "restore of vdi_6.c fails"
echo "x - extracting vdi_7.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_7.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
```

```
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_pmarker(WORD handle, WORD count, WORD pxyarray[])
X{  vdi_macro_c(7, handle, count, pxyarray); }
SHAR_EOF
chmod 0644 vdi_7.c || echo "restore of vdi_7.c fails"
echo "x - extracting vdi_8.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_8.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_gtext(WORD handle, WORD x, WORD y, char *string)
X{
X    int i = 0;
X    ptsin[0] = x;
X    ptsin[1] = y;
X    while (intin[i++] = *string++);
X    contrl[0] = 8;
X    contrl[1] = 1;
X    contrl[3] = --i;
X    contrl[6] = handle;
X    VDI_CALL();
X}
SHAR_EOF
chmod 0644 vdi_8.c || echo "restore of vdi_8.c fails"
echo "x - extracting vdi_9.c (Text)"
```

```
sed 's/^X//' << 'SHAR_EOF' > vdi_9.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID v_fillarea(WORD handle, WORD count, WORD pxyarray[])
X{  vdi_macro_c(9, handle, count, pxyarray); }
SHAR_EOF
chmod 0644 vdi_9.c || echo "restore of vdi_9.c fails"
echo "x - extracting vdi_b.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_b.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XWORD vdi_macro_b(WORD nummer, WORD eingabe1, WORD eingabe2)
X{
```

```
X    contrl[0] = nummer;
X    contrl[1] = 0;
X    contrl[3] = 1;
X    contrl[6] = eingabe1;
X    intin[0]  = eingabe2;
X    VDI_CALL();
X    return intout[0];
X}
X
SHAR_EOF
chmod 0644 vdi_b.c || echo "restore of vdi_b.c fails"
echo "x - extracting vdi_c.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_c.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vdi_macro_c(WORD nummer, WORD eingabe1, WORD eingabe2, WORD eingabe3[])
X{
X    contrl[0] = nummer;
X    contrl[1] = eingabe2;
X    contrl[3] = 0;
X    contrl[6] = eingabe1;
X    pblock[2] = eingabe3;
X    VDI_CALL();
X    pblock[2] = ptsin;
X}
X
X
SHAR_EOF
chmod 0644 vdi_c.c || echo "restore of vdi_c.c fails"
echo "x - extracting vdi_d.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_d.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
```

```
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vdi_macro_d(WORD nummer, WORD handle, WORD eingabe)
X{
X    contrl[0] = nummer;
X    contrl[1] = 0;
X    contrl[3] = 1;
X    contrl[6] = handle;
X    intin[0] = eingabe;
X    VDI_CALL();
X}
X
X
SHAR_EOF
chmod 0644 vdi_d.c || echo "restore of vdi_d.c fails"
echo "x - extracting vdi_e.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_e.c &&
X/*
X *                  GEM library for GNU CC on Atari ST
X *                  Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
```

```
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vdi_macro_e(WORD nummer, WORD handle, LONG pusrcode, LONG *psavcode)
X{
X    *((LONG *) &contrl[7]) = pusrcode;
X    contrl[0] = nummer;
X    contrl[1] = 0;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
X    *psavcode = *((LONG *) &contrl[9]);
X}
X
SHAR_EOF
chmod 0644 vdi_e.c || echo "restore of vdi_e.c fails"
echo "x - extracting vdi_m_a.c (Text)"
sed 's/^X//' << 'SHAR_EOF' > vdi_m_a.c &&
X/*
X *                   GEM library for GNU CC on Atari ST
X *                   Copyright (C) 1989     stefan brueck (sb)
X *
X *    EMail: roeder@sbsvax.uucp
X *
X *    This source is free software; you can redistribute it and/or modify
X *    it under the terms of the GNU General Public License as published by
X *    the Free Software Foundation; either version 1, or (at your option)
X *    any later version.
X *
X *    This source is distributed in the hope that it will be useful,
X *    but WITHOUT ANY WARRANTY; without even the implied warranty of
X *    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
X *    GNU General Public License for more details.
X *
X *    You should have received a copy of the GNU General Public License
X *    along with this program; if not, write to the Free Software
X *    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
X */
X#include "gem-guts.h"
X
XVOID vdi_macro_a(WORD nummer, WORD handle)
X{
X    contrl[0] = nummer;
X    contrl[1] = 0;
X    contrl[3] = 0;
X    contrl[6] = handle;
X    VDI_CALL();
X}
SHAR_EOF
chmod 0644 vdi_m_a.c || echo "restore of vdi_m_a.c fails"
rm -f s2_seq_.tmp
echo "You have unpacked the last part"
```

```
exit 0


From -1057225583732327296
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,3ce41bd624058a9c,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-21 10:42:16 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!csd4.milw.wisc.edu!bionet!ames!
apple!sun-barr!newstop!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i062: postmort -- Analyze post-bomb information
Keywords: 68000
Message-ID: <33917@grapevine.uucp>
Date: 21 Jul 89 17:42:16 GMT
Sender: news@grapevine.uucp
Lines: 347
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: usqb015@liverpool.ac.uk (Mark Powell)
Posting-number: Volume 2, Issue 62
Archive-name: postmort

* Postmortem information analyser
* Copyright 1989 By M.S.Powell (usqb015@uk.ac.liv)
* Permission granted to examine and modify this source for; personal use,
* no profit and as long as this header is left intact.
* Written on Devpac v1.22 by Hisoft

     clr.l     -(sp)
     move #32,-(sp)
     trap #1
     addq.l    #6,sp
     move.l    d0,ssp

     dc.w $a00a

     move #4,-(sp)
     trap #14
     addq.l    #2,sp
     move d0,oldres
     bne.s     resokay
     move #1,-(sp)
     pea  -1.w
     move.l    (sp),-(sp)
     move #5,-(sp)
     trap #14
     lea  12(sp),sp
     move $8246.w,oldcol3
     move $825e.w,$8246.w

resokay   lea  mess0(pc),a6
```

```
     bsr  prmes
     cmp.l    #$12345678,$380.w
     bne  invalid
     moveq    #0,d7
     move.b   $3c4.w,d7
     cmp  #25,d7
     bcc  invalid

     lea  vector(pc),a5
     moveq    #0,d6
     move.b   0(a5,d7),d6
     beq  invalid
     lea  0(a5,d6),a6
     move d7,-(sp)
     bsr  printd7
     move (sp)+,d7
     bsr  prmes
     cmp  #4,d7
     bcc.s    normal

     move #1,type
     lea  mess4(pc),a6
     bsr  prmes
     move $3cc.w,d7
     btst #3,d7
     bne.s    notins
     addq.l   #4,a6
notins   bsr  prmes
     btst #4,d7
     bne.s    read
     addq.l   #5,a6
read bsr  prmes

normal   bsr  cr
     bsr  cr
     lea  $384.w,a3
     lea  $3a4.w,a4
     move.l   $3c8.w,d0
     move.l   $3c0.w,$3c8.w
     move.l   d0,$3c0.w
     moveq    #'0',d7
prreglp  move #'D',d0
     move.l   (a3)+,d3
     bsr  reg
     moveq    #26,d6
prspace  bsr  space
     dbra d6,prspace
     move.l   (a4)+,d3
     cmp  #'7',d7
     bne.s    prnorm
     move #'U',d0
     bsr  prmess3
     bra.s    donereg
```

```
prnorm   move #'A',d0
     bsr  reg
     bsr  cr
     addq #1,d7
     bra.s    prreglp

donereg  lea  mess21(pc),a6
     bsr  prmes
     move.l   $3c8.w,d3
     move.l   $3c0.w,$3c8.w
     move.l   d3,$3c0.w
     bsr  prmes3
     bsr  cr

     lea  $3cc.w,a3
     tst  type
     beq.s    notba

     lea  mess5(pc),a6
     bsr  prmes
     move (a3)+,d6
     and  #7,d6
     moveq    #0,d7
     move d6,d7
     bsr  printd7
     lea  funcode(pc),a5
     moveq    #0,d7
     move.b   0(a5,d6),d7
     lea  0(a5,d7),a6
     bsr  prmes
     bsr  cr
     lea  mess6(pc),a6
     bsr  prmes
     move.l   (a3)+,d3
     bsr  prhex32
     bsr  prmes
     move (a3)+,d3
     bsr  prhex16
     bsr  cr

notba    lea  mess7(pc),a6
     bsr  prmes
     move (a3)+,d5
     move d5,d3
     bsr  prhex16
     bsr  space
     lea  flagdat(pc),a5
     moveq    #6,d6
prflags  cmp  #4,d6
     bne.s    not4
     move     d5,d7
     lsr  #8,d7
     and  #7,d7
```

```
     add  #48,d7
     move d7,-(sp)
     bsr  prchar
not4 move.b   (a5)+,d7
     move.b   (a5)+,d4
     move d4,-(sp)
     btst d7,d5
     bne.s    bitset
     move #32,(sp)
bitset   bsr  prchar
     dbra d6,prflags

     bsr  prmes
     move.l   (a3)+,d3
     bsr  prhex32
     bsr  cr

     lea  $3cc.w,a3
     move.l   $3c0.w,a4
     moveq    #1,d7
linelp   bsr  cr
     move.l   a4,d3
     bsr  prhex32
     bsr  space
     move #'>',-(sp)
     bsr  prchar
     moveq    #7,d6
wordlp   bsr  space
     move (a3)+,d3
     bsr  prhex16
     dbra d6,wordlp
     addq.l   #8,a4
     dbra d7,linelp

     bra.s    out

invalid  lea  mess1(pc),a6
     bsr.s    prmes

out  lea  mess2(pc),a6
     bsr.s    prmes
wait bsr.s    getkey
     cmp  #13,d0
     bne.s    wait

desktop  move oldres(pc),-(sp)
     pea  -1.w
     move.l   (sp),-(sp)
     move #5,-(sp)
     trap #14
     lea  12(sp),sp
     tst  oldres
     bne.s    notlo
```

```
     move oldcol3(pc),$8246.w
notlo    move.l   ssp(pc),-(sp)
     move #32,-(sp)
     trap #1
     addq.l   #6,sp
     clr  -(sp)
     trap #1

prmes    move.b   (a6)+,d0
     beq.s    exit
     move d0,-(sp)
     bsr.s    prchar
     bra.s    prmes

prchar   move.l   (sp)+,return
     pea  $30002
     trap #13
     addq.l   #6,sp
     move.l   return(pc),-(sp)
exit rts

getkey   pea  $10002
     trap #13
     addq.l   #4,sp
     tst  d0
     beq.s    getky
     bsr.s    getky
     bra.s    getkey
getky    pea  $20002
     trap #13
     addq.l   #4,sp
     rts

printd7  move #'#',-(sp)
     bsr.s    prchar
     divu #10,d7
     tst  d7
     beq.s    less10
     add  #48,d7
     move d7,-(sp)
     bsr.s    prchar
less10   swap d7
     add  #48,d7
     move d7,-(sp)
     bsr.s    prchar
     move #32,-(sp)
     bsr.s    prchar
     rts

reg  move d0,-(sp)
     bsr.s    prchar
     move d7,-(sp)
     bsr.s    prchar
```

```
     bsr.s     space
     move #'=',-(sp)
     bsr.s     prchar
     bsr.s     space
     bsr.s     prhex32
     rts

prhex32  move d3,-(sp)
     swap d3
     bsr.s     prhex16
     move (sp)+,d3

prhex16  move d3,-(sp)
     lsr  #8,d3
     bsr.s     prhex8
     move (sp)+,d3

prhex8   move d3,-(sp)
     lsr  #4,d3
     bsr.s     prhex4
     move (sp)+,d3

prhex4   and  #15,d3
     moveq     #$90-1,d4
     addq #1,d4
     abcd d4,d3
     moveq     #$40,d4
     abcd d4,d3
     move d3,-(sp)
     bsr  prchar
     rts

space    move #32,-(sp)
     bsr  prchar
     rts

cr   move #13,-(sp)
     bsr  prchar
     move #10,-(sp)
     bsr  prchar
     rts

prmess3  move d0,-(sp)
     bsr  prchar
     lea  mess3(pc),a6
     bsr  prmes
prmes3   bsr.s     prhex32
     bra.s     cr

oldcol3  dc.w 0

mess0    dc.b 27,'E',27,'Y',32+1,32+23
     dc.b 'Postmortem information analyser v2'
```

```
     dc.b 27,'Y',32+3,32+26
     dc.b 'Copyright 1988 By M.S.Powell'
     dc.b 13,10,10,0
mess1    dc.b '                          '
     dc.b 'Postmortem information invalid',0
mess2    dc.b 13,10,10,'                                  '
     dc.b 'Press RETURN to exit',0
mess21   dc.b 27,'Y',32+15,32+40,'S'
mess3    dc.b 'SP= ',0
mess4    dc.b ' during ',0
     dc.b 'non '
     dc.b 'instruction ',0
     dc.b 'read',0
     dc.b 'write',0
mess5    dc.b 'Function code   = ',0
mess6    dc.b 'Access address  = ',0
     dc.b 27,'Y',32+18,32+40,'Instruction register = ',0
mess7    dc.b 'Status register = ',0
mess8    dc.b '         Program counter      = ',0

ssp  dc.l 0
oldres   dc.w 0
return   dc.l 0
type dc.w 0

flagdat  dc.b 15,'T',13,'S',4,'X',3,'N',2,'Z',1,'V',0,'C'

vector
v    dc.b 0,0,v2-v,v3-v,v4-v,v5-v,v6-v,v7-v,v8-v,v9-v,0,0,0,0,0
     dc.b v15-v,0,0,0,0,0,0,0,0,v24-v
v2   dc.b 'Bus error',0
v3   dc.b 'Address error',0
v4   dc.b 'Illegal instruction',0
v5   dc.b 'Zero divide',0
v6   dc.b 'CHK instruction',0
v7   dc.b 'TRAPV instruction',0
v8   dc.b 'Privilege violation',0
v9   dc.b 'Trace',0
v15  dc.b 'Uninitialised interrupt vector',0
v24  dc.b 'Spurious interrupt',0

funcode
fc   dc.b fc0-fc,fc1-fc,fc2-fc,fc0-fc,fc0-fc,fc5-fc,fc6-fc,fc7-fc
fc0  dc.b '(Undefined, reserved)',0
fc1  dc.b 'User data',0
fc2  dc.b 'User program',0
fc5  dc.b 'Supervisor data',0
fc6  dc.b 'Supervisor program',0
fc7  dc.b 'Interrupt acknowledge',0


From -5621461804848924384
X-Google-Language: ENGLISH,ASCII-7-bit
```

X-Google-Thread: 100493,5e1b9296096dfedf,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-23 13:47:48 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!rutgers!apple!sun-barr!newstop!sun!
grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i063: crunch -- Executable file compressor
Keywords: 68000
Message-ID: <33932@grapevine.uucp>
Date: 23 Jul 89 20:47:48 GMT
Sender: news@grapevine.uucp
Lines: 779
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: usqb015@liverpool.ac.uk (Mark Powell)
Posting-number: Volume 2, Issue 63
Archive-name: crunch

[Binary and docs are in comp.binaries.atari.st. -sg]

```
* Program file Cruncher
* Copyright 1989 By M.S.Powell (usqb015@uk.ac.liv)
* Permission granted to examine and modify this source for; personal use,
* no profit and as long as this header is left intact.
* Written on Devpac v1.22 by Hisoft

* Examine command line

     move.l   4(sp),a5

     lea  129(a5),a5
     cmp.b    #'-',(a5)
     bne.s    nxtfield
     addq.l   #1,a5
chkqual  move.b   (a5)+,d0
     cmp.b    #$d,d0
     beq  invalid
     cmp.b    #' ',d0
     beq.s    nxtfield
     cmp.b    #'D',d0
     bne.s    notd
     move #1,delete
notd cmp.b    #'H',d0
     bne.s    noth
     move #1,hold
noth cmp.b    #'U',d0
     bne.s    notu
     move #1,uncrunch
notu cmp.b    #'I',d0
     bne.s    chkqual
     move #1,ignore
     bra.s    chkqual
```

```
nxtfield
     cmp.b    #' ',(a5)
     bne.s    getname1
     addq.l   #1,a5
     bra.s    nxtfield
getname1
     cmp.b    #$d,(a5)
     beq  invalid
     move.l   a5,sourcenamepnt
getname1loop
     move.b   (a5)+,d0
     cmp.b    #$d,d0
     beq.s    fndend1
     cmp.b    #' ',d0
     bne.s    getname1loop
fndend1  subq.l   #1,a5
     move.l   a5,sourcenameend
     bra.s    intro

invalid  clr  cmndlnokay

* Intro procedure

intro    lea  mainmess(pc),a0
     bsr  print

     tst  cmndlnokay
     bne.s    clokay

     lea  instructions(pc),a0
     bsr  print
     bra  finished

clokay   pea  dta(pc)       set dta buffer
     move #$1a,-(sp)
     trap #1
     addq.l   #6,sp

     clr  -(sp)             Search for source file
     move.l   sourcenamepnt(pc),-(sp)
     move #$4e,-(sp)
     trap #1
     addq.l   #8,sp
     move d0,error
     beq  nextfile

     lea  nomatch(pc),a0
     bsr  print
     bra  finished

* Main loop
```

```
nextfile
     lea  dta+30(pc),a0
     move.l   sourcenameend(pc),a1
     move.l   sourcenamepnt(pc),a2
look_for_name_start
     move.b   -(a1),d0
     cmp.b    #'\',d0
     beq.s    addname
     cmp.b    #':',d0
     beq.s    addname
     cmp.l    a2,a1
     bge.s    look_for_name_start

addname  moveq    #0,d1
     addq.l   #1,a1
anamlp   move.b   (a0)+,d0
     move.b   d0,(a1)+
     beq.s    done_name
     cmp.b    #'.',d0
     bne.s    anamlp
     moveq    #1,d1
     move.l   a1,sourcenameext
     bra.s    anamlp
done_name
     tst  d1
     bne.s    do_it
     move.l   a1,sourcenameext
do_it    bra  do_your_stuff
return
     tst  error
     beq.s    noerror

     tst  ignore
     beq.s    finished

noerror  move #$4f,-(sp)
     trap #1
     addq.l   #2,sp
     tst  d0
     beq.s    nextfile

finished
     tst  hold
     beq.s    nohold

     lea  pressany(pc),a0
     bsr  print
     bsr.s    anykey

nohold   move error(pc),-(sp)        Return error code to calling
     move #$4c,-(sp)         program
     trap #1
```

```
anykey    pea  $20002
     trap #13
     addq.l   #4,sp
     rts

prchar    move d0,-(sp)
     pea  $30002
     trap #13
     addq.l   #6,sp
     rts


* Main loop body

do_your_stuff
     move.l   sp,stack

     lea  reading(pc),a0
     bsr  print
     bsr  printfilename

     move.l   sp,a6
     sub.l    #4096,a6  a6=top of free memory
     move.l   dta+26(pc),d6 d6=file length
     cmp.l    #28,d6
     blt.s    not_a_program

     lea  buffer(pc),a0
     add.l    d6,a0
     cmp.l    a6,a0
     bgt  outofmemory

* Read in file

     lea  buffer(pc),a3
     clr  -(sp)
     move.l   sourcenamepnt(pc),-(sp)
     move #$3d,-(sp)
     trap #1
     addq.l   #8,sp
     tst.l    d0
     bmi  fileerror

     move d0,d7
     pea  (a3)      Read first 2 bytes, to check if it is
     pea  2.w       a valid GEMDOS program
     move d0,-(sp)
     move #$3f,-(sp)
     trap #1
     lea  12(sp),sp
     tst.l    d0
     bmi  fileerror
     cmp  #$601a,(a3)
```

```
     beq.s    a_prog

not_a_program
     lea  notprog(pc),a0
     bsr  print
     bra  exit

a_prog   pea  2(a3)
     move.l   d6,-(sp)
     move d7,-(sp)
     move #$3f,-(sp)
     trap #1
     lea  12(sp),sp
     tst.l    d0
     bmi  fileerror

     move d7,-(sp)
     move #$3e,-(sp)
     trap #1
     addq.l   #4,sp

* Check if file is already crunched

     lea  buffer+$1c(pc),a0
     lea  header+$1c(pc),a1
     move #lengthoftable-header-$1c-1,d0
chklp    cmpm.b   (a0)+,(a1)+
     dbne d0,chklp
     bne  crunchfile

* If file is already crunched, then uncrunch it

     tst  uncrunch
     bne.s    uncrunchit

     lea  crunched(pc),a0
     bsr  print
     bra  exit

uncrunchit
     lea  uncrunching(pc),a0
     bsr  print

     lea  buffer+(lengthoftable-header)(pc),a0
     lea  buffer-lengthoftable(a0),a3
     move.l   (a0)+,a4
     add.l    a3,a4
     move.l   (a0)+,a5
     add.l    a4,a5

     move.l   (a0)+,d0
     move.l   (a3),d1
     and.l    #$ffffff,d1
```

```
     lea  0(a4,d1.l),a0
     move.l   a0,a2
     lea  0(a5,d0.l),a1
     cmp.l    a6,a1
     bgt  outofmemory
     move.l   a1,a6
moveup1  move.b   -(a5),-(a1)
     cmp.l    a5,a0
     bne.s    moveup1
     move.l   a6,a5
     move.l   a1,a6

uncrunchloop1
     move.l   (a3)+,d1
     move.l   d1,d2
     rol.l    #8,d2
     and.l    #$ffffff,d1
     moveq    #0,d3
     move (a3)+,d3
     bne.s    not103
     move.l   #65536,d3
not103   add.l    d0,d1
     sub.l    d3,d0
     lea  0(a4,d1.l),a0
     cmp.l    a6,a0
     beq.s    nomove12
movebitdown1
     move.b   (a6)+,(a2)+
     cmp.l    a6,a0
     bne.s    movebitdown1
nomove12
     subq #1,d3
makeblock1
     move.b   d2,(a2)+
     dbra d3,makeblock1
     tst.l    d0
     bne.s    uncrunchloop1

     move.l   a5,a3
     sub.l    a4,a3

     tst  delete
     beq  writefile
     move.l   sourcenameext(pc),a0
     move.b   #'.',-1(a0)
     move.b   #'U',(a0)+
     move.b   #'C',(a0)+
     move.b   #'R',(a0)+
     bra  writefile

* Crunch a file

crunchfile
```

```
     tst  uncrunch
     beq.s    okaycf

     lea  notcrunched(pc),a0
     bsr  print
     bra  exit

okaycf   lea  crunching(pc),a0
     bsr  print

     move.l   a6,a5
     sub.l    d6,a5
     move.l   a6,a4
     lea  0(a3,d6.l),a3
moveprogup
     move.b   -(a3),-(a4)
     cmp.l    a4,a5
     bne.s    moveprogup

     lea  buffer(pc),a4
     lea  -6(a6),a2
     move.l   2(a4),d7
     add.l    6(a4),d7
     add.l    $a(a4),d7 d7=total original length

     move.l   a5,a0
p1lp moveq    #0,d0
     move.b   (a0),d0
     cmp.b    1(a0),d0
     bne.s    notsam
     cmp.b    2(a0),d0
     bne.s    notsam
     cmp.b    3(a0),d0
     bne.s    notsam
     cmp.b    4(a0),d0
     bne.s    notsam
     cmp.b    5(a0),d0
     bne.s    notsam
     cmp.b    6(a0),d0
     bne.s    notsam

     lea  7(a0),a1
fndend   cmp.l    a6,a1
     bgt.s    endprog
     cmp.b    (a1)+,d0
     beq.s    fndend
endprog  move.l   a0,d1
     sub.l    a5,d1
     ror.l    #8,d0
     or.l d0,d1
     subq.l   #1,a1
     move.l   a1,d2
     sub.l    a0,d2
```

```
nextbit  cmp.l    #65536,d2
     ble.s    lastbit
     move.l   d1,(a4)+
     add.l    #65536,d1
     clr  (a4)+
     sub.l    #65536,d2
     bra.s    nextbit
lastbit  move.l   d1,(a4)+
     move d2,(a4)+
     cmp.l    a5,a4
     bge  outofmemory
     lea  -1(a1),a0

notsam   addq.l   #1,a0
     cmp.l    a2,a0
     blt.s    p1lp

     lea  buffer+4(pc),a3
     cmp.l    a3,a4         Is table empty?
     blt  hopeless
     move.l   a4,-(sp)
     moveq    #0,d1
     move -(a4),d1
     bne.s    not0
     move.l   #65536,d1
not0 move.l   -4(a4),d2
     and.l    #$ffffff,d2
     add.l    d1,d2
     lea  0(a5,d2.l),a2
     cmp.l    a4,a3
     beq.s    lastblock
crunchloop
     move.l   -(a4),d2
     and.l    #$ffffff,d2
     lea  0(a5,d2.l),a1
     moveq    #0,d1
     move -(a4),d1
     bne.s    not01
     move.l   #65536,d1
not01    move.l   -4(a4),d2
     and.l    #$ffffff,d2
     add.l    d1,d2
     lea  0(a5,d2.l),a0
     cmp.l    a1,a0
     beq.s    nomove
removeblock
     move.b   -(a1),-(a2)
     cmp.l    a1,a0
     bne.s    removeblock
nomove   cmp.l    a4,a3
     bne.s    crunchloop

lastblock
```

```
     move.l    -(a4),d1
     and.l     #$ffffff,d1
     lea  0(a5,d1.l),a1
     cmp.l     a5,a1
     beq.s     nomove1
removelastblock
     move.b    -(a1),-(a2)
     cmp.l     a5,a1
     bne.s     removelastblock
nomove1   move.l    a2,a5

     move.l    (sp)+,a4

* Add uncrunch program

done move.l    a4,a3
     move.l    a5,a2     attach crunched prog to end of table
moveprog
     move.b    (a2)+,(a3)+
     cmp.l     a2,a6
     bne.s     moveprog

     move.l    a3,d0
     sub.l     a4,d0
     move.l    d0,lengthofprog

     move.l    a4,d0
     lea  buffer(pc),a2
     sub.l     a2,d0
     move.l    d0,lengthoftable

     moveq     #0,d1
getlengthloop
     addq.l    #4,a2
     moveq     #0,d0
     move (a2)+,d0
     bne.s     not02
     move.l    #65536,d0
not02     add.l     d0,d1
     cmp.l     a2,a4
     bne.s     getlengthloop
     move.l    d1,totallength

     lea  header(pc),a4
     sub.l     a4,a3
     move.l    a3,d0
     add.l     #1023,d0
     and.l     #$fffc00,d0
     add.l     #1023,d6
     and.l     #$fffc00,d6
     cmp.l     d0,d6
     ble  hopeless
     lea  -$1c(a3),a0
```

```
     move.l    a0,2(a4)
     sub.l     a0,d7
     move.l    d7,$a(a4)

     tst  delete
     beq.s     writefile
     move.l    sourcenameext(pc),a0
     move.b    #'.',-1(a0)
     move.b    #'C',(a0)+
     move.b    #'R',(a0)+
     move.b    #'N',(a0)+

writefile
     lea  writing(pc),a0
     bsr  print
     bsr  printfilename

     clr  -(sp)
     move.l    sourcenamepnt(pc),-(sp)
     move #$3c,-(sp)
     trap #1
     addq.l    #8,sp
     tst.l     d0
     bmi  fileerror

     move d0,-(sp)
     pea  (a4)
     pea  (a3)
     move d0,-(sp)
     move #$40,-(sp)
     trap #1
     lea  12(sp),sp
     tst.l     d0
     bmi  fileerror

     move #$3e,-(sp)
     trap #1
     addq.l    #4,sp
     tst.l     d0
     bmi  fileerror

exit move.l    stack(pc),sp
     bra  return



* Error routines

outofmemory
     lea  outmem(pc),a0
     bsr.s     print
     move #-39,error
     bra.s     exit
```

```
fileerror
     move d0,error
     lea  generror(pc),a0
     bsr.s    print
     bra.s    exit

hopeless
     lea  nohope(pc),a0
     bsr.s    print
     bra.s    exit

print    pea  (a0)
     move #9,-(sp)
     trap #1
     addq.l   #6,sp
     rts

printfilename
     move.l   sourcenamepnt(pc),a5   Print file name
prnamlp  move.b   (a5)+,d0
     beq.s    donenameprint
     bsr  prchar
     bra.s    prnamlp
donenameprint
     moveq    #13,d0
     bsr  prchar
     moveq    #10,d0
     bra  prchar

find_target_file_ext
     lea  dta+30(pc),a0
lookext  move.b   (a0)+,d0
     beq.s    fndext
     cmp.b    #'.',d0
     bne.s    lookext

fndext   move.b   #'.',-1(a0)
     rts


* Various data

mainmess
     dc.b 27,'vProgram file cruncher V3.2',13,10,13,10
     dc.b 'Copyright 4/9/1988 By M.S.Powell',13,10,0
instructions
     dc.b 13,10,'Usage:',13,10,13,10
     dc.b 'crunch [h][d][u] <pathname> ',13,10,13,10
     dc.b 'h hold screen before exiting',13,10
     dc.b 'd disable deletion of original file',13,10
     dc.b 'u uncrunch files',13,10,0
```

```
nomatch  dc.b 13,10,'No files match pathname',13,10,0
reading  dc.b 13,10,'Reading ',0
writing  dc.b 'Writing ',0

crunched
     dc.b 'File already crunched',13,10,0
notprog
     dc.b 'Not a GEMDOS program file',13,10,0
notcrunched
     dc.b 'File not crunched',13,10,0
uncrunching
     dc.b 'Uncrunching...',13,10,0
crunching
     dc.b 'Crunching...',13,10,0

outmem   dc.b 'Out of memory',13,10,0
nohope   dc.b 'File uncrunchable',13,10,0
generror
     dc.b 'A file error has occured',13,10,0
pressany
     dc.b 13,10,'Press any key ',0
     even

stack    dc.l 0
error    dc.w 0
cmndlnokay
     dc.w 1

delete   dc.w 0
hold dc.w 0
uncrunch
     dc.w 0
ignore   dc.w 0

sourcenamepnt
     dc.l 0
sourcenameend
     dc.l 0
sourcenameext
     dc.l 0

     even

dta  ds.b 44


* GEMDOS header

header   dc.w $601a
     dc.l 0,0,0,0,0,0
     dc.w $ffff

* Uncrunch routine
```

```
uncrunchprog
     clr.l    -(sp)
     move #32,-(sp)
     trap #1
     addq.l   #6,sp
     move.l   d0,-(sp)

     move $8240.w,-(sp)

     lea  start(pc),a3
     move.l   lengthoftable(pc),a4
     add.l    a3,a4
     move.l   lengthofprog(pc),a5
     add.l    a4,a5

     move.l   totallength(pc),d0
     move.l   (a3),d1
     and.l    #$ffffff,d1
     lea  0(a4,d1.l),a0
     move.l   a0,a2
     lea  0(a5,d0.l),a1
     move.l   a1,a6
moveup   move.b   -(a5),-(a1)
     cmp.l    a5,a0
     bne.s    moveup
     move.l   a6,a5
     move.l   a1,a6

uncrunchloop
     move d5,$8240.w
     add  #$89,d5
     and  #$777,d5
     or   #$333,d5
     move.l   (a3)+,d1
     move.l   d1,d2
     rol.l    #8,d2
     and.l    #$ffffff,d1
     moveq    #0,d3
     move (a3)+,d3
     bne.s    not03
     move.l   #65536,d3
not03    add.l    d0,d1
     sub.l    d3,d0
     lea  0(a4,d1.l),a0
     cmp.l    a6,a0
     beq.s    nomove2
movebitdown
     move.b   (a6)+,(a2)+
     cmp.l    a6,a0
     bne.s    movebitdown
nomove2  subq #1,d3
makeblock
```

```
     move.b    d2,(a2)+
     dbra d3,makeblock
     tst.l     d0
     bne.s     uncrunchloop

* move relocate program to end of text

doneuncrunch
     move.l    a5,d0
     addq.l    #1,d0
     and  #$fffe,d0
     move.l    d0,a1
     move.l    d0,a2
     lea  relocate(pc),a0
     move #start-relocate-1,d0
movereloc
     move.b    (a0)+,(a1)+
     dbra d0,movereloc
     jmp  (a2)

relocate
     move (sp)+,$8240.w
     move.l    8(sp),a0  a0=start of base page
     lea  $100(a0),a1    a1=start of text area
     move.l    a1,a2
     move.l    a0,(a0)+  base address of TPA
     move.l    $436.w,(a0)+   end of TPA+1
     move.l    a1,(a0)+  base address of text
     move.l    2(a4),(a0)+    length of text
     add.l     2(a4),a1
     move.l    a1,(a0)+  base address of data
     move.l    6(a4),(a0)+    length of data
     add.l     6(a4),a1
     move.l    a1,(a0)+  base address of bss
     move.l    $a(a4),(a0)+   length of bss
     move.l    a1,-(sp)
     move.l    $a(a4),-(sp)
     add.l     $e(a4),a1 a1=address of relocation data
     tst  $1a(a4)
     bpl.s     relocpres
     sub.l     a1,a1          a1=0 if no reloc. data
relocpres
     lea  $1c(a4),a4
     move.l    a2,a3          a3=start of text
     clr.b     (a5)
movetextdown
     move.l    (a4)+,(a2)+
     cmp.l     a5,a4
     blt.s     movetextdown
     moveq     #0,d7
     cmp.l     d7,a1
     beq.s     execute
```

```
* Relocate text

     move.l   a3,a6         a6=start of text

     move.l   a6,d0
     move.l   (a1)+,d1
     beq.s    execute
     add.l    d1,a6
     add.l    d0,(a6)
     moveq    #1,d2
     move.l   #$fe,d3
     moveq    #0,d1
reloclp  move.b   (a1)+,d1
     beq.s    execute
     cmp.b    d2,d1
     beq.s    disp
     add.l    d1,a6
     add.l    d0,(a6)
     bra.s    reloclp
disp add.l    d3,a6
     bra.s    reloclp

execute  move.l   (sp)+,d5 d5=length of bss
     move.l   (sp)+,a4  a4=start of bss
     move #32,-(sp)
     trap #1
     addq.l   #6,sp

     lea  start_bss_clearer(pc),a0
     move.l   d5,d6
     add.l    a4,d6
     addq.l   #1,d6
     and  #$fffe,d6
     move.l   d6,a5
     move.l   a5,a6
     move.l   (a0)+,(a5)+
     move.l   (a0)+,(a5)+
     jmp  (a6)

start_bss_clearer
     move.b   d7,(a4)+
     cmp.l    a6,a4
     blt.s    start_bss_clearer
runit    jmp  (a3)
end

lengthoftable dc.l 0
lengthofprog  dc.l 0
totallength   dc.l 0
start
buffer
```

From -5943885313652722694
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,95430246b2a5153c,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-24 08:38:26 PST
Path: utzoo!attcan!utgpu!jarvis.csri.toronto.edu!rutgers!apple!sun-barr!sun!
grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i064: environ -- Set environment variables from AUTO folder
Keywords: shar, 68000
Message-ID: <33942@grapevine.uucp>
Date: 24 Jul 89 15:38:26 GMT
Sender: news@grapevine.uucp
Lines: 200
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: Thomas_E_Zerucha@cup.portal.com (Thomas E. Zerucha)
Posting-number: Volume 2, Issue 64
Archive-name: environ

Environ.prg is a program to set the desktop's environment from the auto
folder using a text file (environ.dat) in the root directory of your boot
disk. This way you can have your environment set for your shell without
long profiles or logins, and can put .rsc files where they can always be
found (on another drive).

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      ENVIRON.DAT
#      ENVIRON.DOC
#      ENVIRON.S
#
sed 's/^X//' << 'SHAR_EOF' > ENVIRON.DAT &&
XPATH=;d:\gemprogs;d:;d:\bin;d:\lang;d:\scripts;d:\usrbin;.
XTEMP=c:\tmp\
XINCLUDE=d:\include\
XSYMB=d:\lang\
XLIB=d:\lib\
XSTARTUP=d:\lib\gemstart
XRUNTIME=d:\lib\gemlib
XSUFF=,.prg,.ttp,.tos
XLIBPATH=d:\mwlib,d:\lang,.
XINCDIR=d:\mwinc\
XTMPDIR=c:\tmp\
XTIMEZONE=EST:0:EDT
XEDITOR=sedt.ttp
XSEDT=d:\include\
XSHELL=csh.tos
```

```
XHOME=d:\
SHAR_EOF
chmod 0644 ENVIRON.DAT || echo "restore of ENVIRON.DAT fails"
sed 's/^X//' << 'SHAR_EOF' > ENVIRON.DOC &&
XENVIRON.PRG
X - set desktop environment (like PATH, HOME, and other things for shells)
X from \AUTO folder.
X
X    Place ENVIRON.PRG in the \AUTO folder.
X
X    Uses ENVIRON.DAT as a standard ASCII text file containing the
X    environment (e.g try "printenv >c:\environ.dat" and use that
X    after fixing the PATH= as shown below).
X
X    Note that GEM can find .RSC files on the first entry in the path,
X    but ignores the first character, e.g. the following works:
X
XNote | the extra semicolon, and that you need semicolons and not commas
X     |         for the PATH to work for the desktop and .rsc files.
X     v
XPATH=;d:\gemprogs;d:;d:\bin;d:\lang;d:\scripts;d:\usrbin;.
XTEMP=c:\tmp\
XINCLUDE=d:\include\
X
X    All my .rsc files (and some configs and such) are in my d:\gemprogs,
X    so I can double click a document (installed as per the desktop
X    dialog, e.g. .DOC=1stword) anywhere on any disk and the program
X    will come up properly, and I can go "gem program doc" from any
X    folder under a shell and everything will work.
X
X    (Tech note: It is documented somewhere that the way GEM searches
X    for the .RSC files (and there is a call that looks for files
X    in the same way for setups, etc.) is to first search the current
X    directory, then the root directory on the disk, then the PATH=
X    value, which defaults to A:, which is why it spins your floppy
X    before it says it can't find the resource.  What my program does
X    is to read in environ.dat (must be in root), change linefeeds
X    to nulls and delete the carriage returns and put nulls at the
X    end, then point the desktop environment to that buffer.)
X
X    What is also useful is that most shells will take these values
X    so you don't have to have a big profile.sh or login.sh files
X    just to set the environment.
X
X    Make sure all lines are of the form SYM=value, and don't put
X    any blank lines or spaces unless needed (see your shell docs).
X
X    Source is included for the curious.
SHAR_EOF
chmod 0644 ENVIRON.DOC || echo "restore of ENVIRON.DOC fails"
sed 's/^X//' << 'SHAR_EOF' > ENVIRON.S &&
X    .text
X*housekeeping
```

```
X    move.l    a7,a5
X    move.l    4(a5),a5 ;compute size for TSR call
X    move.l    $c(a5),d6 ;text
X    add.l     $14(a5),d6     ;data
X*   add.l     $1c(a5),d6     ;bss
X    add.l     #$100,d6  ;basepage
X    add.l     #$40,d6        ;saftey
X*open
X    move.w    #0,-(sp)
X    pea  envfstr
X    move.w    #$3d,-(sp)
X    trap #1
X    addq.l    #8,sp
X    move.w    d0,handl
X    blt  bexit
X*read
X    pea  workar
X    move.l    #8192,-(sp)
X    move.w    handl,-(sp)
X    move.w    #$3f,-(sp)
X    trap #1
X    add.l     #12,sp
X    add.w     d0,d6          ;length of used BSS = length of file
X    move.l    #workar,a0
X    add.w     d0,a0          ;point to end of environment
X    move.b    #0,(a0)        ;put nulls afterword
X    move.b    #0,1(a0)  ;put nulls afterword
X
X;change linefeeds to nulls
X    move.l    #workar,a0
X    move.w    d0,d1
Xzaplflp:
X    cmp.b     #10,(a0)
X    bne.s     notlf
X    move.b    #0,(a0)
Xnotlf:
X    addq.l    #1,a0
X    dbf  d0,zaplflp
X;remove returns
X    move.l    #workar,a0
X    move.l    a0,a1
Xzapcrs:
X    cmp.b     #13,(a0)
X    bne.s     notcr
X    addq.l    #1,a0
X    bra.s     zapcrs
Xnotcr:
X    move.b    (a0)+,(a1)+
X    cmp.b     #0,(a0)
X    dbeq d1,zapcrs
X    move.b    #0,(a1)
X
X    add.w     #$40,d6        ;safety
```

```
X*close
X    move.w   handl,-(sp)
X    move.w   #$3d,-(sp)
X    trap #1
X    addq.l   #4,sp
X
X* alter vector
X    move.l   #doit,-(sp)   ;point to the vector changer code
X    move.w   #38,-(sp) ;we want to do it in supervisor mode!
X    trap #14      ;execute it!
X    addq.l   #6,sp
X* tell everyone we are here
X    pea  msg1
X    move.w   #9,-(sp)
X    trap #1
X    addq.l   #6,sp
X* TSR
X    clr  -(sp)         ;exit but stay
X    move.l   d6,-(sp)
X    move #$31,-(sp)
X    trap #1
X
Xbexit:  move #0,-(sp)
X    trap #1
X* set env upon startup
Xmovenv:
X    move.l   a7,a5
X    move.l   4(a5),a5
X    move.l   a5,a4
Xlinkup:
X    move.l   $24(a4),a4
X    movea.l  $2c(a4),a3
X    cmp.w    #0,(a3)
X    beq.s    setwork
X    move.l   a4,a5
X    bra.s    linkup
Xsetwork:
X
X    move.l   #workar,$2c(a5)
X    move.l   shlsav,a5
X    jmp  (a5)
X***
X* install (in super) vector
Xdoit:   move.l   $4fe,shlsav   ;move
X    move.l   #movenv,$4fe  ;move
X    rts                     ;and return
X***
X************************
X    .data
Xmsg1:        dc.b 'Environment Set\r\n',0
Xenvfstr: dc.b 'ENVIRON.DAT',0
Xhandl:       dc.w 0
Xshlsav:      dc.l 0
```

```
X    .bss
Xworkar:      ds.b 8192
X    .end
SHAR_EOF
chmod 0644 ENVIRON.S || echo "restore of ENVIRON.S fails"
exit 0
```

```
From -6360217430217250623
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,c2a44799ef25f199,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-07-24 08:48:37 PST
Path: utzoo!attcan!utgpu!jarvis.csri.toronto.edu!rutgers!apple!sun-barr!sun!
grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i065: viruskil -- Virus killer
Keywords: 68000
Message-ID: <33945@grapevine.uucp>
Date: 24 Jul 89 15:48:37 GMT
Sender: news@grapevine.uucp
Lines: 1348
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: usqb015@liverpool.ac.uk (Mark Powell)
Posting-number: Volume 2, Issue 65
Archive-name: viruskil

[Documentation is in the binaries posting. -sg]

* Virus killer
* Copyright 1989 by M.S.Powell (usqb015@uk.ac.liv)
* Permission granted to exmaine and modify this source for; personal use,
* no profit and as long as this header is left intact.
* Written on Devpac v1.22 by Hisoft

numrect  equ  6

     move.l   4(sp),a5
     move.l   #end+32000+4096,d0
     move.l   a5,d1
     add.l    d0,d1
     bclr #0,d1
     move.l   d1,sp
     move.l   d0,-(sp)
     move.l   a5,-(sp)
     pea  $4a0000
     trap #1
     lea  12(sp),sp
     tst  d0
     beq.s    okay
```

```
     pea  nomem(pc)
     move #9,-(sp)
     trap #1
     addq.l   #6,sp
     pea  $20002
     trap #13
     addq.l   #4,sp
     bra  exit

okay clr.l    -(sp)
     move #32,-(sp)
     trap #1
     addq.l   #6,sp
     move.l   d0,oldssp

     dc.w $a00a

     move.l   #end,d0
     add.l    #255,d0
     clr.b    d0
     move.l   d0,screen

     move #$19,-(sp)
     trap #1
     addq.l   #2,sp
     cmp  #2,d0
     blt.s    drokay
     clr  d0
drokay   move d0,drive

     move #2,-(sp)
     trap #14
     addq.l   #2,sp
     move.l   d0,physcrn
     move #3,-(sp)
     trap #14
     addq.l   #2,sp
     move.l   d0,logscrn

     movem.l  $8240.w,d0-d7
     movem.l  d0-d7,oldpal

     move #4,-(sp)
     trap #14
     addq.l   #2,sp
     move d0,oldres
     cmp  #2,d0
     beq.s    hires
     clr  d0
hires    move d0,res
     move d0,-(sp)
     move.l   screen(pc),-(sp)
     move.l   (sp),-(sp)
```

```
     move #5,-(sp)
     trap #14
     lea  12(sp),sp

     lea  scrdata(pc),a0
     move.l   screen(pc),a1
     lea  8*160(a1),a1
     move #2560/4-1,d0
movelp   move.l   (a0)+,(a1)+
     dbra d0,movelp

     move #$2700,sr
     move.l   $fa06.w,oldier
     clr.b    $fa07.w
     move.b   #$40,$fa09.w
     bclr #0,$fa0b.w
     bclr #6,$fa0d.w
     bclr #0,$fa0f.w
     bclr #6,$fa11.w
     bset #0,$fa13.w
     bset #6,$fa15.w
     move.l   $118.w,oldkeyb
     move.l   #keybrd,$118.w
     move.l   $120.w,oldtimb
     move.l   #timerb,$120.w
     clr.b    $fa1b.w
     move.b   #1,$fa21.w
     move.b   #8,$fa1b.w
     move.l   $456.w,a0
     move.l   (a0),oldvblq
     move.l   #vbl,(a0)
     move.l   #palette,$45a.w
     move #1,$36.w
     move #1,$38.w
     move #$2300,sr

     lea  keydat(pc),a0
     moveq    #13-1,d0
     bsr  sendikbd

     bsr  wvbl
     lea  icondat(pc),a0
     move.l   screen(pc),a6
     lea  64+32*160(a6),a1
     bsr  drawicon
     lea  16+64*160(a6),a1
     bsr  drawicon
     lea  112+64*160(a6),a1
     bsr  drawicon
     lea  16+120*160(a6),a1
     bsr  drawicon
     lea  112+120*160(a6),a1
     bsr  drawicon
```

```
     lea  drivedata(pc),a0
     lea  48+40*160(a6),a1
     tst  drive
     bne.s    nota
     lea  96(a0),a0
nota bsr  drawdrivenum
     lea  drivedata+96*2(pc),a0
     lea  104+40*160(a6),a1
     tst  drive
     beq.s    notb
     lea  96(a0),a0
notb bsr  drawdrivenum

     bsr  mouseon
     move.l   a6,a5
     lea  mainmess(pc),a0
     bsr  printbl

* Main loop

main bsr  wvbl

     tst  leftbutton
     beq.s    main

     lea  recflag(pc),a0
     lea  routlst(pc),a1
     moveq    #numrect-1,d0
chkrect  tst.b    (a0)+
     bne.s    inone
     addq.l   #4,a1
     dbra d0,chkrect
     bra.s    main

inone    move.l   (a1),a1
     lea  buffer(pc),a6
     move.l   screen(pc),a5
     jsr  (a1)
     move.l   #mousedata,mousedatapnt
waitnb   tst  leftbutton
     bne.s    waitnb
     bra.s    main


drivea   clr  drive
     lea  drivedata+96(pc),a0
     lea  48+40*160(a5),a1
     bsr  mouseoff
     bsr  drawdrivenum
     lea  drivedata+96*2(pc),a0
     lea  104+40*160(a5),a1
     bsr  drawdrivenum
```

```
     bra  mouseon

driveb    cmp  #1,$4a6.w
     beq.s    dbout
     move #1,drive
     lea  drivedata(pc),a0
     lea  48+40*160(a5),a1
     bsr  mouseoff
     bsr  drawdrivenum
     lea  drivedata+96*3(pc),a0
     lea  104+40*160(a5),a1
     bsr  drawdrivenum
     bra  mouseon
dbout     rts

examine   move.l   #busydata,mousedatapnt
     bsr  clrbl
     bsr  read

     lea  $1e(a6),a0
     lea  antivirus(pc),a1
     move #endanti-antivirus-1,d0
chkavlp   move.b   (a0)+,d1
     cmp.b    (a1)+,d1
     bne.s    notant
     dbra d0,chkavlp

     lea  antin(pc),a0
     bra  printbl

notant    move.l   a6,a0
     moveq    #0,d0
     move #256-1,d1
chklp     add  (a0)+,d0
     dbra d1,chklp
     cmp  #$1234,d0
     bne.s    notexec

     lea  exec(pc),a0
     bra  printbl

notexec   lea  noexec(pc),a0
     bra  printbl

install   move.l   #busydata,mousedatapnt
     bsr  clrbl
     bsr  read
     bsr  clrboot
     lea  antivirus(pc),a0
     lea  $1e(a6),a1
     move #endanti-antivirus-1,d0
movealp   move.b   (a0)+,(a1)+
     dbra d0,movealp
```

```
     move #$601c,(a6)
     bsr  chksum
     move d1,(a0)
     bsr  write
     lea  insmess(pc),a0
     bra  printbl

remove    move.l   #busydata,mousedatapnt
     bsr  clrbl
     bsr  read
     bsr  clrboot
     bsr  chksum
     subq #1,d1
     move d1,(a0)
     bsr  write
     lea  remmess(pc),a0
     bra  printbl


* Various routines

sendikbd
keylp     btst #1,$fc00.w
     beq.s     keylp
     move.b    (a0)+,$fc02.w
     dbra d0,keylp
     rts

clrboot   lea  $1e(a6),a0
     move #482/2-1,d0
clrbslp   clr  (a0)+
     dbra d0,clrbslp
     rts

chksum    lea  (a6),a0
     moveq     #0,d1
     move #255-1,d0
chksmlp   add  (a0)+,d1
     dbra d0,chksmlp
     sub  #$1234,d1
     neg  d1
     rts

read pea  1.w
     pea  $10000
     move drive(pc),-(sp)
     subq.l    #4,sp
     pea  (a6)
     move #8,-(sp)
     trap #14
     lea  20(sp),sp
     tst  d0
     bmi  error
```

```
     rts

write    pea  1.w
     pea  $10000
     move drive(pc),-(sp)
     subq.l   #4,sp
     pea  (a6)
     move #9,-(sp)
     trap #14
     lea  20(sp),sp
     tst  d0
     bmi  error
     rts

printbl  bsr  mouseoff
     lea  1+188*160(a5),a1
     lea  152+188*160(a5),a5
     moveq    #0,d1
     move.b   (a0)+,d1
     subq #1,d1
prspace  moveq    #32,d0
     bsr  print
     dbra d1,prspace
prnxtc   move.b   (a0)+,d0
     beq.s    prout
     bsr  print
     bra.s    prnxtc
prout    cmp.l    a1,a5
     beq.s    prex
     move.l   a1,a4
     moveq    #8-1,d0
clrchlp  clr.b    (a4)
     clr.b    2(a4)
     lea  160(a4),a4
     dbra d0,clrchlp
     move.l   a1,d0
     addq.l   #7,d0
     and  #$fff9,d0
     move.l   d0,a1
     bra.s    prout
prex bra  mouseon

print    ext  d0
     cmp  #'A',d0
     blt.s    punct
     cmp  #'Z',d0
     bgt.s    lowerc
     sub  #64,d0
     bra.s    prit
punct    cmp  #',',d0
     bne.s    notcom
     moveq    #53,d0
     bra.s    prit
```

```
notcom   cmp  #'.',d0
     bne.s    notfs
     moveq    #54,d0
     bra.s    prit
notfs    cmp  #'!',d0
     bne.s    notex
     moveq    #55,d0
     bra.s    prit
notex    cmp  #'0',d0
     blt.s    notnum
     cmp  #'9',d0
     bgt.s    notnum
     addq #8,d0
     bra.s    prit
notnum   moveq    #0,d0
     bra.s    prit
lowerc   cmp  #127,d0
     bne.s    notcpyr
     moveq    #66,d0
     bra.s    prit
notcpyr  sub  #70,d0
prit add  d0,d0
     add  d0,d0
     add  d0,d0
     lea  chrset(pc),a3
     lea  0(a3,d0),a3
     move.l   a1,a4
     moveq    #8-1,d0
prntlp   move.b   (a3),(a4)
     move.b   (a3)+,2(a4)
     lea  160(a4),a4
     dbra d0,prntlp
     move.l   a1,d0
     addq.l   #7,d0
     and  #$fff9,d0
     move.l   d0,a1
     rts

clrbl    bsr  mouseoff
     lea  8+188*160(a5),a0
     moveq    #0,d1
     moveq    #18-1,d0
clrbtlp  move.l   d1,(a0)
     move.l   d1,160(a0)
     move.l   d1,320(a0)
     move.l   d1,480(a0)
     move.l   d1,640(a0)
     move.l   d1,800(a0)
     move.l   d1,960(a0)
     move.l   d1,1120(a0)
     addq.l   #8,a0
     dbra d0,clrbtlp
     bra  mouseon
```

```
error    neg  d0
     lea  errors(pc),a0
     move.b   0(a0,d0),d0
     lea  0(a0,d0),a0
     bra  printbl

wvbl move.l   $466.w,d0
waitlp   cmp.l    $466.w,d0
     beq.s    waitlp
     rts

mouseon  movem.l  d0-d7/a0-a2,-(sp)
     bsr  drawm
     movem.l  (sp)+,d0-d7/a0-a2
     move #1,mouse
     rts

mouseoff
     clr  mouse
     movem.l  d0-d7/a0-a2,-(sp)
     bsr  undraw
     movem.l  (sp)+,d0-d7/a0-a2
     rts

drawdrivenum
     moveq    #16-1,d0
drawdnlp
     move.l   a1,a2
     move.l   (a0)+,(a2)+
     move (a0)+,(a2)+
     lea  160(a1),a1
     dbra d0,drawdnlp
     rts

drawicon
     moveq    #32-1,d0
drawilp  move.l   a1,a2
     move.l   (a0)+,(a2)+
     move (a0)+,(a2)+
     addq.l   #2,a2
     move.l   (a0)+,(a2)+
     move (a0)+,(a2)+
     addq.l   #2,a2
     move.l   (a0)+,(a2)+
     move (a0)+,(a2)+
     addq.l   #2,a2
     move.l   (a0)+,(a2)+
     move (a0)+,(a2)+
     lea  160(a1),a1
     dbra d0,drawilp
     rts
```

```
* Vertical blank handler

vbl  clr  $30.w
     subq #1,$38.w
     bne.s    notnewc
     move #3,$38.w
     add  #$90,$36.w
     and  #$771,$36.w
notnewc   move $36.w,$32.w
     move $38.w,$34.w
     move.b   #1,$fa07.w

     tst  mouse
     beq.s    nomouse

     bsr  undraw
     bsr  drawm

     lea  rectdat(pc),a0
     lea  recflag(pc),a1
     move mousex(pc),d1
     move mousey(pc),d2
     moveq    #8-1,d0
rectlp    cmp  (a0),d1
     blt.s    notin
     cmp  2(a0),d2
     blt.s    notin
     cmp  4(a0),d1
     bge.s    notin
     cmp  6(a0),d2
     bge.s    notin
     move.b   #1,(a1)+
     bra.s    nxtrect
notin     clr.b    (a1)+
nxtrect   addq.l   #8,a0
     dbra d0,rectlp

nomouse   rts

timerb    cmp  #31,$30.w
     beq.s    endtimb
     move d0,-(sp)
     move.l   a0,-(sp)
     lea  $8248.w,a0
     move $32.w,d0
     move d0,(a0)+
     addq #1,d0
     move d0,(a0)+
     addq #1,d0
     move d0,(a0)+
     addq #1,d0
     move d0,(a0)+
     addq #1,d0
```

```
     move d0,(a0)+
     addq #1,d0
     move d0,(a0)+
     addq #1,d0
     move d0,(a0)+
     subq #1,d0
     move d0,(a0)+
     subq #1,d0
     move d0,(a0)+
     subq #1,d0
     move d0,(a0)+
     subq #1,d0
     move d0,(a0)+
     subq #1,d0
     move d0,(a0)+
     subq #1,d0
     move d0,$8248.w
     move.l   (sp)+,a0
     move (sp)+,d0
     subq #1,$34.w
     bne.s    noc
     move #3,$34.w
     add  #$90,$32.w
     and  #$771,$32.w
noc  addq #1,$30.w
     clr.b    $fa0f.w
     rte
endtimb  btst #1,$fc00.w
     beq.s    notempt
     move.b   #$d,$fc02.w
notempt  movem.l  d0-d7,-(sp)
     movem.l  palette(pc),d0-d7
     movem.l  d0-d7,$8240.w
     movem.l  (sp)+,d0-d7
     clr.b    $fa07.w
     clr.b    $fa0f.w
     rte

drawm    moveq    #0,d0
     move mousey(pc),d0
     move d0,d1
     add  d0,d0
     add  d0,d0
     add  d1,d0
     lsl  #5,d0
     move mousex(pc),d1
     move d1,d2
     lsr  #1,d1
     and  #$fff8,d1
     add  d1,d0
     move.l   screen(pc),a0
     add.l    d0,a0
     and  #$f,d2
```

```
     lea  trash(pc),a1
     move.l   mousedatapnt(pc),a2
     move.l   a0,(a1)+
     move #200,d3
     sub  mousey(pc),d3
     cmp  #16,d3
     ble.s    fulsize
     moveq    #16,d3
fulsize  subq #1,d3
     move d3,(a1)+
     cmp  #152,d1
     beq.s    rightedge
drawlp   movem.l  (a0),d4-d7
     movem.l  d4-d7,(a1)
     lea  16(a1),a1
     moveq    #0,d0
     move (a2)+,d0
     not  d0
     swap d0
     lsr.l    d2,d0
     not.l    d0
     move.l   a0,a4
     swap d0
     and  d0,(a4)+
     and  d0,(a4)+
     and  d0,(a4)+
     and  d0,(a4)+
     swap d0
     and  d0,(a4)+
     and  d0,(a4)+
     and  d0,(a4)+
     and  d0,(a4)+
     moveq    #0,d0
     move (a2)+,d0
     swap d0
     lsr.l    d2,d0
     or   d0,8(a0)
     swap d0
     or   d0,(a0)
     moveq    #0,d0
     move (a2)+,d0
     swap d0
     lsr.l    d2,d0
     or   d0,10(a0)
     swap d0
     or   d0,2(a0)
     lea  160(a0),a0
     dbra d3,drawlp
     rts
rightedge
     movem.l  (a0),d4-d7
     movem.l  d4-d7,(a1)
     lea  16(a1),a1
```

```
     move (a2)+,d0
     not  d0
     lsr  d2,d0
     not  d0
     move.l   a0,a4
     and  d0,(a4)+
     and  d0,(a4)+
     and  d0,(a4)+
     and  d0,(a4)+
     move (a2)+,d0
     lsr  d2,d0
     or   d0,(a0)
     move (a2)+,d0
     lsr  d2,d0
     or   d0,2(a0)
     lea  160(a0),a0
     dbra d3,rightedge
     rts

undraw   lea  trash(pc),a0
     move.l   (a0)+,a1
     move (a0)+,d4
undrlp   movem.l  (a0)+,d0-d3
     movem.l  d0-d3,(a1)
     lea  160(a1),a1
     dbra d4,undrlp
     rts


* Keyboard handler

keybrd   movem.l  d0-d1/a0,-(sp)

again    move.b   $fc00.w,d1
     btst #7,d1
     beq  keyout
     btst #0,d1
     beq  keyo

     moveq    #0,d0
     move.b   $fc02.w,d0
     tst  length
     beq.s    nopack

     move.l   bufpnt(pc),a0
     move.b   d0,(a0)+
     move.l   a0,bufpnt
     subq #1,length
     bne  keyo

     lea  keybuf(pc),a0
     cmp.b    #$f7,(a0)+
     bne.s    keyo
```

```
     addq.l    #1,a0
     move (a0)+,mousex
     move (a0)+,mousey
     bra.s     keyo

nopack    cmp  #$f6,d0
     blt.s     akey

     lea  keybuf(pc),a0
     move.b    d0,(a0)+
     move.l    a0,bufpnt
     sub  #$f6,d0
     move.b    lengths(pc,d0),d0
     move d0,length
     bra.s     keyo

lengths   dc.b 7,5,2,2,2,2,6,2,1,1

akey move d0,keys
     cmp  #$74,d0
     bne.s     notlon
     move #1,leftbutton
     bra.s     keyo
notlon    cmp  #$f4,d0
     bne.s     keyo
     clr  leftbutton

keyo btst #5,d1
     beq.s     keyout

     move.b    $fc02.w,d0

keyout    btst #4,$fa01.w
     beq  again

     movem.l   (sp)+,d0-d1/a0
     bclr #6,$fa11.w
     rte

* Return to desktop

desktop   move #$2700,sr
     move.l    oldier(pc),$fa06.w
     move.l    oldkeyb(pc),$118.w
     move.l    $456.w,a0
     move.l    oldvblq(pc),(a0)
     move #$2300,sr

     lea  keydat1(pc),a0
     moveq     #3-1,d0
     bsr  sendikbd
```

```
     movem.l   oldpal(pc),d0-d7
     movem.l   d0-d7,$8240.w

     move oldres(pc),-(sp)
     move.l    physcrn(pc),-(sp)
     move.l    logscrn(pc),-(sp)
     move #5,-(sp)
     trap #14
     lea  12(sp),sp
     move.l    oldssp(pc),-(sp)
     move #32,-(sp)
     trap #1
     addq.l    #6,sp
exit clr  -(sp)
     trap #1

* The anti-virus!

antivirus
     move.l    #$5000777,$8240.w

     move #4,-(sp)
     trap #14
     addq.l    #2,sp
     move d0,d7
     beq.s     lores

     lea  starti(pc),a0
     lea  $140.w,a1
     move.l    a1,a2
     move #endi-starti-1,d0
moveilp  move.b    (a0)+,(a1)+
     dbra d0,moveilp
     move.l    $456.w,a0
     move.l    a2,28(a0)

lores     move.l    $44e.w,a0
     lea  92*160+2*8(a0),a0
     tst  d7
     beq.s     nhires
     subq.l    #8,a0
nhires    lea  mess(pc),a1
     lea  chars-8(pc),a2
prvflp    moveq     #0,d0
     move.b    (a1)+,d0
     beq.s     donevf
     add  d0,d0
     add  d0,d0
     add  d0,d0
     lea  0(a2,d0),a3
     move.l    a0,a4
     moveq     #8-1,d0
linelp    move.b    (a3)+,d1
```

```
     moveq    #8-1,d2
expndlp  lsl.b    #1,d1
     move sr,d4
     tst  d7
     bne.s    hires0
     roxl #1,d3
     move d4,sr
     roxl #1,d3
     bra.s    nxtshft
hires0   roxl.l   #1,d3
     move d4,sr
     roxl.l   #1,d3
     move d4,sr
     roxl.l   #1,d3
     move d4,sr
     roxl.l   #1,d3
nxtshft  dbra d2,expndlp
     tst  d7
     bne.s    hires1
     move d3,(a4)
     move d3,160(a4)
     bra.s    addisp
hires1   move.l   d3,(a4)
     move.l   d3,80(a4)
     move.l   d3,160(a4)
     move.l   d3,240(a4)
addisp   lea  320(a4),a4
     dbra d0,linelp
     tst  d7
     bne.s    hires2
     addq.l   #8,a0
     bra.s    prvflp
hires2   addq.l   #4,a0
     bra.s    prvflp

donevf   rts

starti   move.l   $44e.w,a0
     tst.l    32000-4(a0)
     beq.s    stilclr
     move #1,$8240.w
     move.l   $456.w,a0
     clr.l    28(a0)
stilclr  rts
endi

mess dc.b 1,2,3,4,5,10,6,3,7,7,10,8,2,5,9,0

chars    dc.b 195,102,102,102,102,102,60,24
     dc.b 48,0,24,12,12,12,12,0
     dc.b 0,192,124,118,96,96,96,0
     dc.b 0,0,230,102,102,110,59,0
     dc.b 0,0,62,96,60,134,124,0
```

```
     dc.b 14,24,24,24,62,24,24,0
     dc.b 0,0,60,102,126,96,62,0
     dc.b 12,6,6,62,102,102,58,0
     dc.b 0,192,102,108,120,108,230,0
     dc.b 0,0,0,0,0,0,0,0
endanti

* Various data


name dc.b 'virus.pic',0

nomem    dc.b 27,'EOut of memory',13,10,13,10
     dc.b 'Press any key',0

exec dc.b 6,'Executable, possible virus!',0
noexec   dc.b 6,'Boot sector not executable',0
antin    dc.b 5,'Disk has anti-virus installed',0

insmess  dc.b 9,'Anti virus installed',0
remmess  dc.b 4,'Executable boot sector removed',0

mainmess
     dc.b 10,127,' 1988 M.S.Powell',0

e
errors   dc.b e0-e,e0-e,e0-e,e0-e,e1-e,e0-e,e0-e,e0-e,e2-e,e0-e,e3-e
     dc.b e4-e,e0-e,e5-e,e0-e,e0-e,e0-e,e0-e
e0   dc.b 12,'General error',0
e1   dc.b 14,'CRC error',0
e2   dc.b 11,'Sector not found',0
e3   dc.b 13,'Write error',0
e4   dc.b 14,'Read error',0
e5   dc.b 9,'Disk write protected',0

     even

mousedatapnt
     dc.l mousedata
mousedata
     dc.w $7F,$FF80,0,$7F,$FF80,$7F00,$FF,$FF00
     dc.w $7E00,$1FF,$FE00,$7C00,$3FF,$FC00,$7800,$1FF
     dc.w $FE00,$7400,$8FF,$F700,$6200,$1C7F,$E380,$4100
     dc.w $3E3F,$C1C0,$80,$FF1F,$E0,$40,$FF8F,$70
     dc.w $20,$FFC7,$38,$10,$FFE3,$1C,8,$FFF1
     dc.w 14,4,$FFFB,4,0,$FFFF,0,0
busydata
     dc.w $8003,$7FFC,$3FE0,1,$FE3E,$3FE0,0,$FE3F
     dc.w $3FE0,0,$FE3F,$3FE0,0,$FE3F,$3FE0,0
     dc.w $FFFF,$3FE0,0,$FFFF,0,0,$FFFF,0
     dc.w 0,$FFFF,0,0,$FFFF,0,0,$FFFF
     dc.w 0,0,$FFFF,0,0,$FFFF,0,0
     dc.w $FFFF,0,0,$FFFF,0,$8001,$7FFE,0
```

```
mouse     dc.w 0
keys dc.w 0
leftbutton     dc.w 0
mousex    dc.w 0
mousey    dc.w 0
length    dc.w 0
bufpnt    dc.l 0
keybuf    ds.b 8
keydat    dc.b 7,4,$e,0,0,0,0,0,9,1,$3f,0,$c7
keydat1   dc.b 7,0,8

chrset    dc.b 0,0,0,0,0,0,0,0
     dc.b 0,24,60,102,102,126,102,0
     dc.b 0,252,102,124,102,102,252,0
     dc.b 0,62,102,96,96,102,60,0
     dc.b 0,252,102,102,102,102,252,0
     dc.b 0,254,98,120,96,98,254,0
     dc.b 0,254,98,120,96,96,240,0
     dc.b 0,124,196,192,222,204,124,0
     dc.b 0,102,102,126,102,102,102,0
     dc.b 0,255,153,24,24,153,255,0
     dc.b 0,30,12,12,12,204,120,0
     dc.b 0,247,108,120,120,108,246,3
     dc.b 0,240,96,96,96,98,254,0
     dc.b 0,198,238,254,214,198,198,0
     dc.b 0,231,118,126,110,102,231,0
     dc.b 0,60,102,102,102,102,60,0
     dc.b 0,252,102,102,124,96,240,0
     dc.b 0,60,102,102,102,108,54,1
     dc.b 0,252,102,102,124,108,246,3
     dc.b 0,62,96,60,6,6,124,0
     dc.b 0,255,153,24,24,24,60,0
     dc.b 0,102,102,102,102,102,60,0
     dc.b 0,102,102,102,60,60,24,0
     dc.b 0,198,198,214,254,238,198,0
     dc.b 0,231,102,60,60,102,231,0
     dc.b 0,231,98,52,24,24,60,0
     dc.b 0,254,140,24,48,98,254,0
     dc.b 0,0,120,204,204,204,118,0
     dc.b 224,96,124,102,102,102,252,0
     dc.b 0,0,62,102,96,102,60,0
     dc.b 14,12,124,204,204,204,118,0
     dc.b 0,0,60,102,126,96,62,0
     dc.b 28,48,48,48,124,48,48,48
     dc.b 0,6,60,102,102,62,96,126
     dc.b 224,96,124,102,102,102,247,0
     dc.b 24,0,56,24,24,24,60,0
     dc.b 6,0,14,6,6,6,102,60
     dc.b 224,96,108,120,120,108,246,3
     dc.b 56,24,24,24,24,24,60,0
     dc.b 0,0,254,219,219,219,219,0
     dc.b 0,0,220,102,102,102,247,0
```

```
     dc.b 0,0,60,102,102,102,60,0
     dc.b 0,0,252,102,102,124,96,240
     dc.b 0,0,126,204,204,124,12,14
     dc.b 0,0,220,102,96,96,240,0
     dc.b 0,0,62,96,60,6,124,0
     dc.b 24,24,126,24,24,24,14,0
     dc.b 0,0,238,102,102,102,59,0
     dc.b 0,0,231,98,116,56,16,0
     dc.b 0,0,198,214,254,124,108,0
     dc.b 0,0,102,60,24,60,102,0
     dc.b 0,0,102,102,102,62,6,124
     dc.b 0,0,126,76,24,50,126,0
     dc.b 0,0,0,0,0,24,24,48
     dc.b 0,0,0,0,0,24,24,0
     dc.b 0,24,24,24,24,0,24,0
     dc.b 0,60,102,110,118,102,60,0
     dc.b 0,24,56,24,24,24,60,0
     dc.b 0,60,102,12,48,98,126,0
     dc.b 0,126,76,24,12,102,60,0
     dc.b 0,12,28,60,108,126,12,0
     dc.b 0,126,98,124,6,102,60,0
     dc.b 0,60,96,124,102,102,60,0
     dc.b 0,126,70,12,24,48,48,0
     dc.b 0,60,102,60,102,102,60,0
     dc.b 0,60,102,62,6,12,56,0
     dc.b %00111100
     dc.b %01000010
     dc.b %10011001
     dc.b %10100001
     dc.b %10100001
     dc.b %10011001
     dc.b %01000010
     dc.b %00111100

rectdat  dc.w 96,40,111,55
     dc.w 208,40,223,55

     dc.w 32,64,95,95
     dc.w 224,64,287,95
     dc.w 32,120,95,151
     dc.w 224,120,287,151

recflag  dc.b 0,0,0,0,0,0

palette  dc.w $000,$105,$333,$555,$777,$040,$020,$600
     dc.w $625,$500,$666,$000,$551,$200,$231,$666

routlst  dc.l drivea,driveb,examine,install,remove,desktop
drive    dc.w 0

oldier   dc.l 0
oldkeyb  dc.l 0
oldtimb  dc.l 0
```

```
oldssp   dc.l 0
oldvblq  dc.l 0
physcrn  dc.l 0
logscrn  dc.l 0
oldres   dc.w 0
res  dc.w 0
screen   dc.l 0

drivedata
     dc.w 0,0,$FFFF,0,0,$8001,0,0
     dc.w $8001,$180,0,$8181,$180,$240,$83C1,$3C0
     dc.w 0,$83C1,$3C0,$420,$87E1,$660,0,$8661
     dc.w $7E0,$810,$8FF1,$C30,$240,$8E71,$C30,$1008
     dc.w $9C39,$1818,$420,$9C39,$810,$1008,$9819,0
     dc.w 0,$8001,0,0,$8001,0,0,$FFFF
     dc.w 0,$FFFF,0,$7FFE,$FFFF,0,$4002,$C003
     dc.w $3FFC,$5E7A,$DE7B,$3FFC,$5E7A,$DE7B,$3DBC,$5C3A
     dc.w $DC3B,$3FFC,$5C3A,$DC3B,$3BDC,$599A,$D99B,$3FFC
     dc.w $581A,$D81B,$37EC,$53CA,$D3CB,$3DBC,$53CA,$D3CB
     dc.w $2FF4,$47E2,$C7E3,$3BDC,$57EA,$D7EB,$2FF4,$4002
     dc.w $C003,$3FFC,$7FFE,$FFFF,0,0,$FFFF,0
     dc.w 0,0,$FFFF,0,0,$8001,0,0
     dc.w $8001,$FC0,$20,$8FE1,$1FE0,$10,$9FF1,$1830
     dc.w $640,$9E71,$1830,$4C0,$9CF1,$1FE0,$10,$9FF1
     dc.w $1FF0,0,$9FF1,$1818,$620,$9E39,$1818,$460
     dc.w $9C79,$1FF0,$8,$9FF9,$FE0,$10,$8FF1,0
     dc.w 0,$8001,0,0,$8001,0,0,$FFFF
     dc.w 0,$FFFF,0,$7FFE,$FFFF,0,$4002,$C003
     dc.w $3FFC,$503A,$D03B,$3FFC,$401A,$C01B,$3FFC,$47CA
     dc.w $C7CB,$3FFC,$47CA,$C7CB,$3FFC,$401A,$C01B,$3FFC
     dc.w $400A,$C00B,$3FFC,$47E2,$C7E3,$3FFC,$47E2,$C7E3
     dc.w $3FFC,$400A,$C00B,$3FFC,$501A,$D01B,$3FFC,$4002
     dc.w $C003,$3FFC,$7FFE,$FFFF,0,0,$FFFF,0

icondat
     dc.w 0,0,0,0,0,0,0,0
     dc.w 0,0,0,0,0,0,0,0
     dc.w 0,0,0,0,0,0,0,0
     dc.w 0,0,0,0,0,0,0,0
     dc.w 0,0,0,0,0,0,0,0
     dc.w 0,0,0,0,0,0,0,0
     dc.w 0,0,0,0,0,0,0,0
     dc.w 0,0,0,0,0,0,0,0
     dc.w 0,0,0,0,0,0,0,0
     dc.w 14,15,$30,$AAAA,$FFFF,0,$B000,$F800
     dc.w 0,0,0,0,$33,$73,12,$D555
     dc.w $FFFF,0,$5600,$FF00,0,0,0,0
     dc.w 12,$7C,3,$7AAA,$7FFF,$8000,$AAC0,$FFE0
     dc.w 0,0,0,0,1,$7F,0,$8FFF
     dc.w $8FFF,$7000,$FFF8,$FFFC,0,0,0,0
     dc.w 0,$7F,0,$33FF,$F3FF,$C00,$FFFE,$FFFF
     dc.w 0,0,0,0,0,$7F,0,$C7F
     dc.w $FC7F,$380,$FFFF,$FFFF,0,$C000,$E000,0
```

```
dc.w 0,$7F,0,$19F,$FF9F,$60,$FFFF,$FFFF
dc.w 0,$F000,$F800,0,0,$7F,0,$60
dc.w $FFE0,$1F,0,0,$FFFF,0,$200,$FC00
dc.w 0,$7F,0,$1F,$FFFF,0,$FFFF,$FFFF
dc.w 0,$FE00,$FE00,0,0,$7F,0,7
dc.w $FFFF,0,$FFFF,$FFFF,0,$FE00,$FE00,0
dc.w 0,$3F,0,7,$FFFF,0,$FF00,$FFFF
dc.w 0,$1E00,$FE00,0,0,$1F,0,6
dc.w $FFFF,0,0,$FFFF,0,$600,$FE00,0
dc.w 0,7,0,5,$FFFE,0,$C000,0
dc.w $3FFF,$3600,$E00,$C000,0,1,0,5
dc.w $FFFE,0,$FFFF,$3FFF,0,$F600,$CE00,0
dc.w 0,0,0,4,$7FFF,0,0,$FFFF
dc.w 0,$E00,$FE00,0,0,0,0,7
dc.w $1FFF,0,$FC00,$FFFF,0,$7E00,$FE00,0
dc.w 0,0,0,7,$7FF,0,$F800,$F800
dc.w $7FF,$FE00,$FE00,$3000,0,0,0,7
dc.w $1FF,0,$F001,$FFFF,0,$FE00,$FE00,$6000
dc.w 0,0,0,7,$7F,0,$FFFF,$FFFF
dc.w 0,$FC00,$FE00,0,0,0,0,1
dc.w $1F,0,$FFFF,$FFFF,0,$F800,$FC00,0
dc.w 0,0,0,0,3,0,0,$FFFF
dc.w 0,0,$F800,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,$600,0,0,$FF00,0
dc.w $FF00,0,0,0,0,0,0,$F03
dc.w 0,3,$81C0,$7E00,$FFC0,0,0,0
dc.w 0,0,0,$1F86,1,7,$1E0,$8000
dc.w $81E0,0,0,0,0,0,0,$3FD8
dc.w 6,$1E,$70,$4008,$78,0,0,0
dc.w 0,0,0,$7FF0,9,$18,$18,$E000
dc.w $18,0,0,0,0,0,0,$FFF0
dc.w 7,$30,8,$F004,12,0,0,0
dc.w 1,0,0,$FFE2,$17,$70,12,$F802
dc.w 14,0,0,0,3,0,0,$FF62
dc.w $8B,$64,4,$FC02,6,0,0,0
dc.w 7,0,0,$FEC0,$12B,$E4,6,$FE01
dc.w 7,0,0,0,15,0,0,$FC84
dc.w $357,$C8,2,$F709,3,0,0,0
dc.w $1E,0,1,$F884,$757,$C8,$C0A,$E389
dc.w 3,0,0,0,$3C,0,3,$7C8C
dc.w $35F,$80C0,$1E0A,$C1C9,3,0,0,0
dc.w $78,0,7,$3E88,$15F,$C0C0,$3F12,$80F1
dc.w 11,0,0,0,$F0,0,15,$1F88
dc.w $5F,$E0C0,$7F92,$71,11,0,0,0
dc.w $1E0,0,$1F,$F80,$5E,$F0C0,$FFE2,$29
dc.w $13,0,0,0,$1C0,0,$3F,$7C1
```

```
dc.w $2C,$F8E0,$FFE6,$21,$17,0,0,0
dc.w $380,0,$7F,$3E3,8,$FC60,$FFE6,$30
dc.w 6,0,0,0,$100,0,$FF,$1E7
dc.w $10,$FE70,$FF8E,$60,14,0,0,0
dc.w 0,$100,$FF,$F7,0,$FF30,$FF0E,$E0
dc.w 14,0,0,0,0,$80,$7F,$79
dc.w 0,$FF98,$FE1E,$180,$1E,0,0,0
dc.w 0,$40,$3F,$26,$18,$FFDE,$7C7F,$200
dc.w $7F,0,0,0,0,$20,$1F,$1B
dc.w 4,$FFE7,$81FF,0,$81FF,$8000,0,$8000
dc.w 0,$10,15,$38,3,$FFC3,$FF01,$FE
dc.w $FFFF,$E000,0,$E000,0,8,7,$70
dc.w 0,$FF80,0,$FF01,$FF01,$7000,$8000,$F000
dc.w 0,4,3,$E0,0,$FF00,0,0
dc.w 0,$3800,$C000,$F800,0,2,1,$1C0
dc.w 0,$FE00,0,0,0,$1E00,$6000,$7E00
dc.w 0,1,0,$380,0,$FC00,0,0
dc.w 0,$700,$3800,$3F00,0,0,0,$700
dc.w $8000,$7800,0,0,0,$380,$C00,$F80
dc.w 0,0,0,$E00,$4000,$3000,0,0
dc.w 0,$180,$640,$7C0,0,0,0,$1C00
dc.w $2000,0,0,0,0,0,$3C0,$3C0
dc.w 0,0,0,$1800,0,0,0,0
dc.w 0,0,$180,$180,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w $3FF,0,0,$FFFF,0,0,$C000,0
dc.w 0,0,0,0,$7FF,0,0,$FFFF
dc.w 0,0,$E000,0,0,0,0,0
dc.w $3FF,$7FF,0,$FFFF,$F000,0,$E000,0
dc.w 0,0,0,0,$200,$600,$1FF,$1FFF
dc.w $1000,$E000,$E000,0,0,0,0,0
dc.w $200,$600,$1FF,$1FFF,$1000,$E000,$E000,0
dc.w 0,0,0,0,$200,$600,$1FF,$1FFF
dc.w $1000,$E000,$E000,0,0,0,0,0
dc.w $200,$600,$1FF,$1FFF,$1000,$E000,$E00C,12
dc.w 12,0,0,0,$200,$600,$1FF,$1FFF
dc.w $1000,$E000,$E01C,$1C,$1C,0,0,0
dc.w $200,$600,$1FF,$1FFF,$1000,$E000,$E03C,$3C
dc.w $3C,0,0,0,$200,$600,$1FF,$1FFF
dc.w $1000,$E000,$E07C,$7C,$7C,0,0,0
dc.w $200,$600,$1FF,$1F80,$107F,$E000,$FF,$E0FF
dc.w $FF,$EEE0,$EEE0,$EEE0,$200,$600,$1FF,$1F00
dc.w $10FF,$E000,$1FF,$E1FF,$1FF,$EEE0,$EEE0,$EEE0
dc.w $200,$600,$1FF,$1F00,$10FF,$E000,$3FF,$E3FF
dc.w $3FF,$EEE0,$EEE0,$EEE0,$200,$600,$1FF,$1F00
dc.w $10FF,$E000,$7FF,$E7FF,$7FF,$EEE0,$EEE0,$EEE0
dc.w $200,$600,$1FF,$1F00,$10FF,$E000,$3FF,$E3FF
dc.w $3FF,$EEE0,$EEE0,$EEE0,$200,$600,$1FF,$1F00
dc.w $10FF,$E000,$1FF,$E1FF,$1FF,$EEE0,$EEE0,$EEE0
dc.w $200,$600,$1FF,$1F00,$10FF,$E000,$FF,$E0FF
```

```
dc.w $FF,$EEE0,$EEE0,$EEE0,$200,$600,$1FF,$1F00
dc.w $10FF,$E000,$7C,$E07C,$7C,0,0,0
dc.w $200,$600,$1FF,$1F7F,$1080,$E000,$C03C,$203C
dc.w $3C,0,0,0,$200,$600,$1FF,$1F7F
dc.w $1080,$E000,$C01C,$201C,$1C,0,0,0
dc.w $200,$600,$1FF,$1F7F,$1080,$E000,$C00C,$200C
dc.w 12,0,0,0,$200,$600,$1FF,$1F7F
dc.w $1080,$E000,$C000,$2000,0,0,0,0
dc.w $200,$600,$1FF,$1F00,$10FF,$E000,0,$E000
dc.w 0,0,0,0,$200,$600,$1FF,$1F80
dc.w $107F,$E000,0,$E000,0,0,0,0
dc.w $200,$600,$1FF,$1FFF,$1000,$E000,$E000,0
dc.w 0,0,0,0,$3FF,$7FF,0,$FFFF
dc.w $F000,0,$E000,0,0,0,0,0
dc.w $7FF,0,0,$FFFF,0,0,$C000,0
dc.w 0,0,0,0,$3FF,0,0,$FFFF
dc.w 0,0,$8000,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w $7FF,0,0,$FFFF,0,0,$8000,0
dc.w 0,0,0,0,$FFF,0,0,$FFFF
dc.w 0,0,$C000,0,0,0,0,0
dc.w $7FF,$FFF,0,$FFFF,$E000,0,$C000,0
dc.w 0,0,0,0,$400,$C00,$3FF,$3FFF
dc.w $2000,$C000,$C000,0,0,0,0,0
dc.w $400,$C00,$3FF,$3FFF,$2000,$C000,$C000,0
dc.w 0,0,0,0,$400,$C00,$3FF,$3FFF
dc.w $2000,$C000,$C000,0,0,0,0,0
dc.w $400,$C00,$3FF,$3FFF,$2000,$C000,$C000,0
dc.w 0,0,0,0,$400,$C00,$3FF,$3FFF
dc.w $2000,$C000,$C000,0,0,$1800,$1800,$1800
dc.w $400,$C00,$3FF,$3FFF,$2000,$C000,$C000,0
dc.w 0,$1C00,$1C00,$1C00,$400,$C00,$3FF,$3FFF
dc.w $2000,$C000,$C000,0,0,$1E00,$1E00,$1E00
dc.w $400,$C00,$3FF,$3F00,$20FF,$C000,0,$C000
dc.w 0,$1F00,$1F00,$1F00,$400,$C00,$3FF,$3E00
dc.w $21FF,$C000,$3BB,$C3BB,$3BB,$FF80,$FF80,$FF80
dc.w $400,$C00,$3FF,$3E00,$21FF,$C000,$3BB,$C3BB
dc.w $3BB,$FFC0,$FFC0,$FFC0,$400,$C00,$3FF,$3E00
dc.w $21FF,$C000,$3BB,$C3BB,$3BB,$FFE0,$FFE0,$FFE0
dc.w $400,$C00,$3FF,$3E00,$21FF,$C000,$3BB,$C3BB
dc.w $3BB,$FFF0,$FFF0,$FFF0,$400,$C00,$3FF,$3E00
dc.w $21FF,$C000,$3BB,$C3BB,$3BB,$FFE0,$FFE0,$FFE0
dc.w $400,$C00,$3FF,$3E00,$21FF,$C000,$3BB,$C3BB
dc.w $3BB,$FFC0,$FFC0,$FFC0,$400,$C00,$3FF,$3E00
dc.w $21FF,$C000,$3BB,$C3BB,$3BB,$FF80,$FF80,$FF80
dc.w $400,$C00,$3FF,$3EFF,$2100,$C000,$8000,$4000
dc.w 0,$1F00,$1F00,$1F00,$400,$C00,$3FF,$3EFF
dc.w $2100,$C000,$8000,$4000,0,$1E00,$1E00,$1E00
```

```
dc.w $400,$C00,$3FF,$3EFF,$2100,$C000,$8000,$4000
dc.w 0,$1C00,$1C00,$1C00,$400,$C00,$3FF,$3EFF
dc.w $2100,$C000,$8000,$4000,0,$1800,$1800,$1800
dc.w $400,$C00,$3FF,$3E00,$21FF,$C000,0,$C000
dc.w 0,0,0,0,$400,$C00,$3FF,$3F00
dc.w $20FF,$C000,0,$C000,0,0,0,0
dc.w $400,$C00,$3FF,$3FFF,$2000,$C000,$C000,0
dc.w 0,0,0,0,$7FF,$FFF,0,$FFFF
dc.w $E000,0,$C000,0,0,0,0,0
dc.w $FFF,0,0,$FFFF,0,0,$8000,0
dc.w 0,0,0,0,$FFF,0,0,$FFFF
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,0,0,0,0,0
dc.w 0,0,0,$100,$C0,$1C0,0,0
dc.w 0,0,0,0,0,0,0,$F00
dc.w $10C0,$1FC0,0,0,0,0,0,0
dc.w 0,1,1,$FF00,$C0,$FFC0,0,0
dc.w 0,0,0,0,3,4,7,$FF00
dc.w $DE,$FFDE,0,0,0,0,0,0
dc.w 7,0,7,$FF3C,$C2,$FFC2,0,0
dc.w 0,0,0,0,7,0,7,$FF3C
dc.w $C2,$FFC2,0,0,0,0,0,0
dc.w 7,0,7,$FF3C,$C2,$FFC2,0,0
dc.w 0,0,0,0,7,0,7,$FF3C
dc.w $C2,$FFC2,$800,$800,$800,0,0,0
dc.w 7,0,7,$FF3C,$C2,$FFC2,$1800,$1800
dc.w $1800,0,0,0,7,0,7,$FF3C
dc.w $C2,$FFC2,$3800,$3800,$3800,0,0,0
dc.w 7,0,7,$FF3C,$C2,$FFC2,$7800,$7800
dc.w $7800,0,0,0,7,0,7,$FF3C
dc.w $C2,$FFC2,$F800,$F800,$F800,0,0,0
dc.w 7,0,7,$FF3D,$C1,$FFC1,$FFEE,$FFEE
dc.w $FFEE,$E000,$E000,$E000,7,0,7,$F33B
dc.w $4C3,$FBC3,$FFEE,$FFEE,$FFEE,$E000,$E000,$E000
dc.w 7,0,7,$F137,$6C7,$FBC7,$FFEE,$FFEE
dc.w $FFEE,$E000,$E000,$E000,7,0,7,$F13F
dc.w $6CF,$FBCF,$FFEE,$FFEE,$FFEE,$E000,$E000,$E000
dc.w 7,0,7,$F137,$EC7,$F3C7,$FFEE,$FFEE
dc.w $FFEE,$E000,$E000,$E000,7,0,7,$F13B
dc.w $EC3,$FFC3,$FFEE,$FFEE,$FFEE,$E000,$E000,$E000
dc.w 7,0,7,$FF3D,$C1,$FFC1,$FFEE,$FFEE
dc.w $FFEE,$E000,$E000,$E000,7,0,7,$FF3C
dc.w $C2,$FFC2,$F800,$F800,$F800,0,0,0
dc.w 7,0,7,$FF3C,$C2,$FFC2,$7800,$7800
dc.w $7800,0,0,0,7,0,7,$FF3C
dc.w $C2,$FFC2,$3800,$3800,$3800,0,0,0
dc.w 7,0,7,$FF3C,$C2,$FFC2,$1800,$1800
dc.w $1800,0,0,0,7,0,7,$FF3C
dc.w $C2,$FFC2,$800,$800,$800,0,0,0
dc.w 7,0,7,$FF3C,$C2,$FFC2,0,0
dc.w 0,0,0,0,3,4,7,$FF3C
```

```
      dc.w $C2,$FFC2,0,0,0,0,0,0
      dc.w 0,1,1,$FF00,$DE,$FFDE,0,0
      dc.w 0,0,0,0,0,0,0,$1F00
      dc.w $60C0,$7FC0,0,0,0,0,0,0
      dc.w 0,0,0,$F00,$10C0,$1FC0,0,0
      dc.w 0,0,0,0,0,0,0,$300
      dc.w $4C0,$7C0,0,0,0,0,0,0
      dc.w 0,0,0,$100,$C0,$1C0,0,0
      dc.w 0,0,0,0,0,0,0,0
      dc.w $40,$40,0,0,0,0,0,0

scrdata   dc.l 0,0,$7FF0000,0,$3FF0000,0,$F8000000,0,$70000,0
      dc.l $C7C00000,0,$1FC00000,0,0,0,0,0,0,0,0,$10000
      dc.l 0,$F3F00000,0,$7F00000,0,$1FC0000,0,$3F0000,0
      dc.l $80000000,0,0,0,0,0,0,0,0,0,$C010000,$3FE0000
      dc.l $83FF0000,$1FF0000,$F8000000,$F0000000,$C0003,$30000
      dc.l $6C608380,$83800000,$1FC00F80,$F800000,0,0,0,0,0,0
      dc.l 0,0,$30000,0,$FFF8E1E0,$E1E0,$4100000,$3E003E0
      dc.l $3FE0000,$F800F8,$60001F,$1F001F,$C0000000,0,0,0,0,0
      dc.l 0,0,0,0,$18000000,$7FF0000,$C3FF0000,$1FF0000
      dc.l $F8000000,$F0000000,$80007,$70000,$2820C7C0,$C7C00000
      dc.l $1FC00F80,$F800000,0,0,0,0,0,0,0,0,$30001,1
      dc.l $FFF8F3F0,$F3F0,$4100000,$3E003E0,$3FE0000,$1FC01FC
      dc.l $40003F,$3F003F,$40008000,$80008000,0,0,0,0,0,0,0
      dc.l 0,$10200000,$FDF0000,$43FF0000,$801F0000,$F8000000,0
      dc.l $80007,$70000,$2820C7C0,$C7C00000,$1FC00000,0,0,0,0
      dc.l 0,0,0,0,0,$30001,1,$FFF8F7E0,$F7E0,$7F00000,0
      dc.l $3FE0000,$FC00FC,$60001F,$1F001F,$40008000,$80008000,0
      dc.l 0,0,0,0,0,0,0,$10500000,$F8F0000,$403F0000
      dc.l $801F0000,$80000000,0,$80007,$70000,$2820C7C0,$C7C00000
      dc.l $3F800000,0,$1FFE0000,0,$3E30000,0,$E03F0000,0
      dc.l $FC000000,0,$30001,1,$FFF0F7C0,$F7C0,$FE00000,0
      dc.l $1FE0000,$7C007C,$30000F,$F000F,$400F8000,$80008000
      dc.l $FE030000,0,$FFC00000,0,0,0,0,0,$105F0000,$F800000
      dc.l $C03F0000,$1F0000,$80000000,0,$80007,$70000,$2820C7C0
      dc.l $C7C00000,$7FC01F00,$1F000000,$30030000,$FFC,$7F70000
      dc.l $1C1,$F060001F,$C01F,$600F800,$F800,$30001,1,$FFE0FF80
      dc.l $FF80,$18300000,$7C007C0,$FE0000,$7C007C,$10000F,$F000F
      dc.l $401F8007,$80078007,$FF06FC00,$FC01FC00,$600000,$FF800000
      dc.l 0,0,0,0,$103F0000,$FC00000,$3F0000,$1F0000,$80000000
      dc.l 0,$80007,$70000,$2820C7C0,$C7C00000,$7FC03F80,$3F800000
      dc.l $20010000,$1FFE,$87F70000,$3E3,$F0C0003F,$E03F,$200FC00
      dc.l $FC00,$30001,1,$FFC0FF00,$FF00,$10100000,$FE00FE0
      dc.l $FE0000,$7C007C,$10000F,$F000F,$403F800F,$800F800F
      dc.l $FF84FE00,$FE03FE00,$300000,$FFC00000,0,0,0,0
      dc.l $18010000,$7FE0000,$803F0000,$1F0000,$80000000,0,$80007
      dc.l $70000,$2820C7C0,$C7C00000,$7FC01F80,$1F800000,$20400000
      dc.l $1FBF,$87F70000,$3E3,$F081007E,$E07E,$FE000000,0,$30001
      dc.l 1,$FF80FE00,$FE00,$18100000,$7E007E0,$FE0000,$7C007C
      dc.l $10000F,$F000F,$403F801F,$801F801F,$FF841F00,$1F031F00
      dc.l $8100000,$F7E00000,0,0,0,0,$C000000,$3FF0000
      dc.l $C03F0000,$1F0000,$80000000,0,$80007,$70000,$2820C7C0
      dc.l $C7C00000,$3FC00F80,$F800000,$20E10000,$1F1E,$87F70000
```

```
     dc.l $3E3,$F083007C,$E07C,$F8000000,0,$30001,1,$FF80FC00
     dc.l $FC00,$C100000,$3E003E0,$FE0000,$7C007C,$10000F,$F000F
     dc.l $403F801F,$801F801F,$FF841F00,$1F031F00,$1C300000
     dc.l $E3C00000,0,0,0,0,$7E00000,$1F0000,$403F0000
     dc.l $801F0000,$80000000,0,$80007,$70000,$3820C7C0,$C7C00000
     dc.l $1FC00F80,$F800000,$20BF0000,$1F00,$7F70000,$3E3
     dc.l $F0C0003F,$E03F,$C00F000,$F000,$30001,1,$FFC0FF00,$FF00
     dc.l $4100000,$3E003E0,$FE0000,$7C007C,$10000F,$F000F
     dc.l $403F801F,$801F801F,$FF84FF00,$FF03FF00,$17E00000
     dc.l $E0000000,0,0,0,0,$1FD00000,$F0000,$403F0000
     dc.l $801F0000,$80000000,0,$C0003,$30000,$1060EF80,$EF800000
     dc.l $1FC00F80,$F800000,$20800000,$1F00,$7F70000,$3E3
     dc.l $F060001F,$E01F,$600F800,$F800,$30001,1,$FFE0FF80,$FF80
     dc.l $4100000,$3E003E0,$FE0000,$7C007C,$10000F,$F000F
     dc.l $403F801F,$801F801F,$FF84FE00,$FE03FE00,$10000000
     dc.l $E0000000,0,0,0,0,$10500000,$F8F0000,$403F0000
     dc.l $801F0000,$80000000,0,$40003,$30000,$40FF80,$FF800000
     dc.l $3FE00F80,$F800000,$20800000,$1F00,$7FF0000,$3E3
     dc.l $F03F0000,$E000,$82007C00,$7C00,$30001,1,$FFF0F7C0,$F7C0
     dc.l $C180000,$3E003E0,$1FF0000,$7C007C,$30000F,$F000F
     dc.l $603F801F,$801F801F,$FF040000,$30000,$10000000,$E0000000
     dc.l 0,0,0,0,$10200000,$FDF0000,$403F0000,$801F0000
     dc.l $80000000,0,$60001,$10000,$C0FF00,$FF000000,$7FF01FC0
     dc.l $1FC00000,$20800000,$1F00,$7FF0000,$3F7,$F0FF0000,$E000
     dc.l $200FC00,$FC00,$30001,1,$FFF8F7E0,$F7E0,$180C0000
     dc.l $7F007F0,$3FF0000,$FE00FE,$8060001F,$1F001F,$303FC01F
     dc.l $C01FC01F,$FE048000,$80038000,$10000000,$E0000000,0,0
     dc.l 0,0,$18000000,$7FF0000,$C03F0000,$1F0000,$80000000,0
     dc.l $30000,0,$180FE00,$FE000000,$7FF03FE0,$3FE00000
     dc.l $20800000,$1F00,$7FF0000,$1FF,$F080007F,$E07F,$600F800
     dc.l $F800,$30001,1,$FFF8F3F0,$F3F0,$10040000,$FF80FF8
     dc.l $3FF0000,$1FF01FF,$8040003F,$3F003F,$103FE00F,$E00FE00F
     dc.l $FE04FC00,$FC03FC00,$10000000,$E0000000,0,0,0,0
     dc.l $C010000,$3FE0000,$803F0000,$E0000,$80000000,0,$10000,0
     dc.l $83007C00,$7C000000,$7FF01FC0,$1FC00000,$31800000,$E00
     dc.l $3FF0000,$FF,$F0C0003F,$C03F,$C00F000,$F000,$30000,0
     dc.l $FFF8E1E0,$E1E0,$180C0000,$7F007F0,$3FF0000,$FE00FE
     dc.l $8060001F,$1F001F,$301FC007,$C007C007,$FE06F800,$F801F800
     dc.l $30000000,$C0000000,0,0,0,0,$7FF0000,0,$1F0000,0,0
     dc.l 0,0,0,$FE000000,0,$3FE00000,0,$1F000000,0,$1FF0000
     dc.l 0,$E07F0000,0,$F8000000,0,$10000,0,$F3F00000,0
     dc.l $FF80000,0,$1FF0000,0,$3F0000,0,$E00F0000,0,$FC030000
     dc.l 0,$E0000000,0,0,0

oldpal
buffer   equ  oldpal+32
trash    equ  buffer+512
end  equ  trash+300


From 1620325054011490770
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,4c8e952c7953e160,start
```

X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-08-07 16:09:03 PST
Path: utzoo!utgpu!watmath!att!dptg!rutgers!cs.utexas.edu!sun-barr!newstop!sun!
sunfedcomm!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i066: norad -- Earth satellite tracking program
Keywords: shar, GFA-BASIC
Message-ID: <34051@grapevine.uucp>
Date: 7 Aug 89 23:09:03 GMT
Sender: news@grapevine.uucp
Lines: 1063
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: logajan@ns.network.com (John Logajan)
Posting-number: Volume 2, Issue 66
Archive-name: norad

Below are two files generically named NORAD Release V1.0.
NORAD is a GFA Basic (2.0) implementation of a real-time
multiple earth satellite tracking program. Earth satellites
are graphically depicted against a map of earth's continents.
It is a very pretty display in my humble opinion.

A sample orbital elements file containing 98 satellites is
included, but I suggest you make an effort to get ahold of
the most recent released elements if you want to accurately
track satellites.

Thanks -- John Logajan.

```
#       This is a shell archive.
#       Remove everything above and including the cut line.
#       Then run the rest of the file through sh.
#----cut here-----cut here-----cut here-----cut here----#
#!/bin/sh
# shar: Shell Archiver
#       Run the following text with /bin/sh to create:
#       norad.lst
#       orbitals.dat
cat << \SHAR_EOF > norad.lst
Rem      N.O.R.A.D. V1.0 The graphic realtime satellite tracking program.
Rem                     ATARI ST -- GFA BASIC 2.0
Rem               Please customize the six lines below.
Rem
Tlocal=-5             ! Your time difference from GMT (UTC). i.e. -5 = CDT
L9=45.0709            ! Your latitude, degrees.fraction-of-degrees.
W9=93.1567            ! Your longitude, degrees.fraction. i.e +values for USA
H9=980                ! Your altitude above sea-level. (Feet)
El$="C:ORBITALS.DAT"  ! Path to orbital elements file. (NASA Predictions)
Ms=500                ! Maximum number of satellites. (To reserve space)
Rem
Rem Orbital calculation subroutines from the AMSAT Orbital Prediction Program
```

```
Rem (C) 1980 by Dr. T. A. Clark W3IWI, 6388 Guilford Road, Clarksville MD
Rem 21209 USA.  "Permission granted for non-commercial use provided credit is
Rem given to author, AMSAT and ORBIT magazine.  See ORBIT issue 6, March/
Rem April 1981.  Ported to ATARI ST BASIC by Scott Tilley (1/24/87.)
Rem
Rem Graphic interface and world map by John Logajan, 4248 Hamline, Arden Hills
Rem MN 55112 USA. 8/7/89  Graphic interface and map code are released into
Rem the public domain.
Rem
Dim C(3,2),Lvt%(200),S$(Ms),Y3(Ms),T0(Ms),Mmr(Ms),I0(Ms),O0(Ms),E0(Ms)
Dim W0(Ms),M0(Ms),N0(Ms),K0(Ms),A0(Ms),Xx%(Ms),Yy%(Ms)
H9=H9*12/39.37
Repeat
  Repeat
    Deftext 3,16,0,32
    Text 130,25,0,"Welcome to N.O.R.A.D."
    Deftext 1,0,0,13
    Text 105,45,0,"Your clock said the time was:  "+Date$+"  "+Time$
    Deftext 2,0,0,13
    Text 100,61,0,"Please be sure it is set to the EXACT time and date!"
    Text 175,160,0,"Press the HELP key for instructions."
    Text 80,186,0,"- Point at satellite of interest and click to get its info -"
    A$="Want to set/reset the clock?"
    Alert 2,A$,1,"No need|SetClock",But
    If But=2
      Cls
      Print "Enter date (or RETURN to skip)"
      Print
      Print "Use the format  month/day/year."
      Print "Be sure to use two digits (leading zeroes) for each."
      Print "Otherwise the system will ignore the new date."
      Print
      Print Date$;
      Input "  Date ";D$
      Cls
      Print "Enter local time (or RETURN to skip)"
      Print
      Print "Use the format  hour:minute:second."
      Print "Be sure to use two digits (leading zeroes) for each."
      Print "Otherwise the system will ignore the new time."
      Print
      Print Time$;
      Input "  Time ";T$
      Settime T$,D$
    Endif
    Cls
  Until But=1
  Y=Val(Right$(Date$,4))-1900
  If Y<89
    Print "          Is it really the year ";1900+Y;" ?"
    Print "I haven't made the program so it can work in years before 1989,"
    Print "so I am going to force you to set your clock/calender."
    Alert 1," ",1,"continue",But
```

```
    Cls
  Endif
Until Y>=89
If Y/4=Int(Y/4)
  F9=1
Else
  F9=0                    ! leap-year flag
Endif
D=Val(Mid$(Date$,4,2))
M=Val(Left$(Date$,2))
Rem Calculate day of year
Data 0,31,59,90,120,151,181,212,243,273,304,334
For I=1 To 12
  Read D9x
  If I=M
    D9=D9x
  Endif
Next I
D8=D+D9
If M>2
  D8=D8+F9
Endif
T$=Time$
Tstart=Timer
H=Val(Left$(T$,2))
S=Val(Right$(T$,2))
M=Val(Mid$(T$,4,2))
T7=D8+H/24+M/1440+S/86400-Tlocal/24
R0=6378.135                           ! physical constants
F=298.25
G0=7.5369793E+13
G1=1.0027379093
P2=2*Pi
P0=Pi/180
F=1/F
Rem  Greenwich Sidereal Time at New Year's
Yt%=0
For Y1=89 To Y
  Yd%=365
  If Y1/4=Int(Y1/4)
    Yd%=366
  Endif
  Yt%=Yt%+Yd%
Next Y1
G2=Frac(Yt%*G1+0.27676779)
Rem  Routine to evaluate OBSERVER's GEOCENTRIC Coordinates, where
Rem   X-axis=GREENWICH, Y-axis goes thru India, Z-axis=North Pole
L8=L9*P0
S9=Sin(L8)
C9=Cos(L8)                                  ! Initial GEOCENTRIC coordinates
S8=Sin(-W9*P0)
C8=Cos(W9*P0)                               ! W9= West longitude
Rem Now to account for flattened Earth
```

```
R9=R0*(1-(F/2)+(F/2)*Cos(2*L8))+H9/1000      ! H9=height in meters
Rem Now to make L8 the GEOCENTRIC latitude
L8=Atn((1-F)^2*S9/C9)
Z9=R9*Sin(L8)
X9=R9*Cos(L8)*C8
Y9=R9*Cos(L8)*S8
Rem   ********** get Parameters for Satellites *********
Print "Reading orbital elements from file: ";El$
Print
On Error Gosub Missing
Open "I",#1,El$
Sp%=0
On Error Gosub Baddy
Do
  Exit If Eof(#1)
  Line Input #1,Ln$
  Exit If Eof(#1)
  Line Input #1,L1$
  Exit If Eof(#1)
  Line Input #1,L2$
  While Left$(L1$,1)<>"1" Or Left$(L2$,1)<>"2"
    Ln$=L1$
    L1$=L2$
    Exit If Eof(#1)
    Line Input #1,L2$
  Wend
  Exit If Eof(#1)
  Inc Sp%
  S$(Sp%)="[ "+Left$(Ln$,11)+" "+Mid$(L1$,3,5)+" ]"
  Y3(Sp%)=Val(Mid$(L1$,19,2))
  T0(Sp%)=Val(Mid$(L1$,21,12))
  If Y3(Sp%)<Y                      ! Correct for "old" orbital elements
    Yt%=0
    For Y1=Y3(Sp%) To Y-1
      Yd%=365
      If Y1/4=Int(Y1/4)
        Yd%=366
      Endif
      Yt%=Yt%+Yd%
    Next Y1
    T0(Sp%)=T0(Sp%)-Yt%
    Y3(Sp%)=Y
  Endif
  Mmr(Sp%)=Val(Mid$(L1$,34,10))
  I0(Sp%)=Val(Mid$(L2$,9,8))
  O0(Sp%)=Val(Mid$(L2$,18,8))
  E0(Sp%)=Val("."+Mid$(L2$,27,7))
  W0(Sp%)=Val(Mid$(L2$,35,8))
  M0(Sp%)=Val(Mid$(L2$,44,8))
  N0(Sp%)=Val(Mid$(L2$,53,11))
  K0(Sp%)=Val(Mid$(L2$,64,5))
  A0(Sp%)=0
  Xx%(Sp%)=-1
```

```
  Yy%(Sp%)=-1
  Print Sp%;Tab(5);S$(Sp%)
  If Sp%>Ms
    Print "Oops! Not enough room reserved for all satellite elements."
    Print "Increase the variable -Ms- to a higher number and try again."
    Edit
  Endif
  If Inp?(2)
    If Inp(2)=27
      Edit
    Endif
  Endif
Loop
On Error
If Sp%=0
  Gosub Missing
Endif
Close #1
Mxsp%=Sp%-1
Rem   ******** Here follows the actual computation loop *********
Cls
Setcolor 0,0,0,0  ! For color 0 (black -- background)
Setcolor 1,0,0,7  ! For color 2 (blue -- trails)
Setcolor 3,7,0,0  ! For color 1 (red -- satellites/text/arrow)
Setcolor 2,3,2,0  ! For color 3 (brown -- world map)
Deffill 3,1
White!=False
Hilite!=False
Facts!=False
Clkon!=False
Mapon!=True
D$=Chr$(248)
Gosub Map
Do
  For Sp%=1 To Mxsp%
    T=T7+(Timer-Tstart)/17280000
    Gosub Fnc
    Gosub Fnm
    Gosub Fnk
    Gosub Fnx
    Color 2
    If Hilite!
      If Sp%=Sr%
        Color 1
      Endif
    Endif
    Plot Xx%(Sp%),Yy%(Sp%)
    Yy%(Sp%)=1.11111111*(90-L5)
    Wx5=W5+180
    If Wx5>=360
      Wx5=Wx5-360
    Endif
    Xx%(Sp%)=1.77777777*(360-Wx5)
```

```
Color 1
Plot Xx%(Sp%),Yy%(Sp%)
If Facts!
  Swap Sp%,Sr%
  T=T7+(Timer-Tstart)/17280000
  Gosub Fnc
  Gosub Fnm
  Gosub Fnk
  Gosub Fnx
  Print "AZ=";Int(A9);D$;Tab(8);"EL=";Int(E9);D$;Tab(16);"ALT=";
  Print Int((R-R0)*0.6215);"m";Tab(28);"LT=";Int(L5);D$;Tab(36);
  Print "LN=";Int(W5);D$;Tab(44);"ORB#=";K;Tab(56);S$(Sp%);Chr$(13);
  Swap Sp%,Sr%
Endif
If Mousek<>0
  Xm%=Mousex
  Ym%=Mousey
  Hilite!=False
  Facts!=False
  Pause 5
  Mk%=Mousek
  If Mk%=3
    Cls
    Gosub Map
  Else
    Print Spc(78);Chr$(13);
    If Mk%=1 Or Mk%=2
      Found!=False
      For Sr%=0 To Mxsp%
        If ((Xx%(Sr%)-Xm%) And &HFFE)=0 And ((Yy%(Sr%)-Ym%) And &HFFE)=0
          Found!=True
          Print Chr$(7);    ! Sound bell!
          Pause 50
        Endif
        Exit If Found!
      Next Sr%
      If Sr%=0
        Found!=False
        Print "That's not a satellite, that's you!";Chr$(13);
      Endif
      If Mk%=1
        Facts!=Found!
      Endif
      If Mk%=2
        Hilite!=Found!
      Endif
    Endif
  Endif
Endif
If Inp?(2)
  Ex$=Inkey$
  If Ex$=Chr$(27)      ! Esc key quits.
    Gosub Done
```

```
      Endif
      If Ex$=Chr$(0)+"b"   ! Help key
        Gosub Help
      Endif
      If Ex$=Chr$(0)+"G"   ! Clr/Home key clears screen
        Cls
        Facts!=False
        Hilite!=False
        Gosub Map
      Endif
      If Ex$=Chr$(0)+";"   ! F1 - toggel background
        White!=Not White!
        If White!
          Setcolor 0,7,7,7
          Setcolor 1,0,0,0
          Setcolor 2,5,6,3
        Else
          Setcolor 0,0,0,0
          Setcolor 1,0,0,7
          Setcolor 3,7,0,0
          Setcolor 2,3,2,0
        Endif
      Endif
      If Ex$=Chr$(0)+"<"   ! F2 - toggel world map drawing
        Mapon!=Not Mapon!
        If Mapon!
          Print "Map will be redrawn after next CLR/HOME    ";Chr$(13);
        Else
          Print "Map will not be redrawn after next CLR/HOME";Chr$(13);
        Endif
        Pause 25
      Endif
      If Ex$=Chr$(0)+"="   ! F3 - toggel clock
        Clkon!=Not Clkon!
        If Not Clkon!
          Text 480,199,0,"                    "
          Fill 481,198
        Else
          Deftext 1,0,0,6
          Text 480,199,0,Time$+" "+Date$
        Endif
      Endif
    Endif
  Next Sp%
  If Clkon!
    Text 480,199,0,Time$+" "+Date$
  Endif
Loop
End
Procedure Missing
  Print "The data file is missing or empty!  You need the orbital elements"
  Print "data for each satellite -- available on-line from the Celestial BBS,"
  Print " @ 513-427-0674, Fairborn Ohio, TS Kelso -- sysop."
```

```
  Print
  Print "Or, if you have the file, make sure you change the path in the first"
  Print "few lines of this program to point to it!  Right now the path is set"
  Print "to point at ";El$
  Stop
Return
Procedure Baddy
  Print "The data file ";El$;" has problems."
  Print "The most likely problem is that the line at the end of the file is"
  Print "not terminated by a carriage return (CR) and/or line feed (LF)."
  Print "Please correct the file and try it again!"
  Stop
Return
Procedure Done
  Color 1
  Setcolor 0,7,7,7
  Setcolor 3,0,0,0
  Setcolor 1,7,0,0
  Setcolor 2,0,7,0
  Edit
Return
Procedure Help
  Alt$="Aim mouse arrow to select:"
  Alt$=Alt$+"|Satellite's data -left button"
  Alt$=Alt$+"|Highlight trail  -right button"
  Alt$=Alt$+"|Clear screen     -both buttons"
  Alert 0,Alt$,1,"Resume|MoreHelp|STOP!",But
  If But=2
    Alt$="F1  - Toggel background color"
    Alt$=Alt$+"|F2  - World-map off/on"
    Alt$=Alt$+"|Esc - Stop program"
    Alt$=Alt$+"|Clr/Home - Clear screen"
    Alert 0,Alt$,1,"Resume|MoreHelp",But
    If But=2
      Alt$="To sight actual satellite:"
      Alt$=Alt$+"|AZ: 0=North, 90=East 180S/270W"
      Alt$=Alt$+"|EL: 0=Horizon, 90=Straight up"
      Alert 0,Alt$,1,"Resume|MoreHelp",But
      If But=2
        Alt$="ALT: Altitude(Miles/sea-level)"
        Alt$=Alt$+"|LT/LN: Latitude/Longitude"
        Alt$=Alt$+"|ORB#: Current revolution #"
        Alt$=Alt$+"|- xx - Description & Sat #"
        Alert 0,Alt$,1,"Resume|MoreHelp",But
        If But=2
          Alt$="Misc functions:"
          Alt$=Alt$+"|F3 - Show clock on/off"
          Alt$=Alt$+"|ALT+HELP - Print screen"
          Alt$=Alt$+"|Hint: F1 before printing!"
          Alert 0,Alt$,1,"EndHelp",But
        Endif
      Endif
    Endif
```

```
  Else
    If But=3
      Gosub Done
    Endif
  Endif
Return
Rem ********** ORBIT DETERMINATION ROUTINES ******************
Procedure Fnc
  Rem Routine to initialize the C(J,K) coordinate rotation matrix
  Rem   and other parameters associated with the orbital elements.
  A0(Sp%)=((G0/(N0(Sp%)^2))^(1/3))
  E2=1-E0(Sp%)^2
  E1=Sqr(E2)
  Q0=M0(Sp%)/360+K0(Sp%)                        ! Q0=initial orbit phase
  Rem -----Account for nodal effects due to lumpy gravity field due to
  Rem        the flattened, oblate spheroid shape of Earth
  K2=9.95*((R0/A0(Sp%))^3.5)/(E2^2)
  Rem Update elements to current epoch and evaluate their SIN/COSs
  S1=Sin(I0(Sp%)*P0)
  C1=Cos(I0(Sp%)*P0)                       ! I0=inclination
  O=O0(Sp%)-(T-T0(Sp%))*K2*C1
  S0=Sin(O*P0)
  C0=Cos(O*P0)                          ! O = R.A.A.N. (deg)
  W=W0(Sp%)+(T-T0(Sp%))*K2*(2.5*(C1^2)-0.5)
  S2=Sin(W*P0)
  C2=Cos(W*P0)                          ! W = arg of perigee
  Rem Set up coordinate rotation matrix for the current orbit
  C(1,1)=+(C2*C0)-(S2*S0*C1)
  C(1,2)=-(S2*C0)-(C2*S0*C1)
  C(2,1)=+(C2*S0)+(S2*C0*C1)
  C(2,2)=-(S2*S0)+(C2*C0*C1)
  C(3,1)=+(S2*S1)
  C(3,2)=+(C2*S1)
Return
Rem ------------------------------------------------------------------
Procedure Fnm
  Rem ---- Function to evaluate M = MEAN ANOMALY in 0-2pi range----
  Rem          K = perigee passage counter
  Rem          used by Phase-3 telemetry on General Beacon
  Q=N0(Sp%)*(T-T0(Sp%))+Q0
  K=Int(Q)
  M=(Q-K)*P2
Return
Rem ------------------------------------------------------------------
Procedure Fnk
  Rem Routine to solve KEPLER's equation, given M
  Rem  returns satellite's GEOCENTRIC coordinates
  E=M+E0(Sp%)*Sin(M)+0.5*(E0(Sp%)^2)*Sin(2*M)      !     initial trial value
  Rem  interation loop to solve Kepler's transcental equation-------
  Do
    S3=Sin(E)
    C3=Cos(E)
    R3=1-E0(Sp%)*C3
```

```
    M1=E-E0(Sp%)*S3
    M5=M1-M
    Exit If Abs(M5)<1.0E-06
    E=E-M5/R3
  Loop
  Rem  Now get satellite's XYZ coordinates------------
  X0=A0(Sp%)*(C3-E0(Sp%))
  Y0=A0(Sp%)*E1*S3
  R=A0(Sp%)*R3                          ! in the plane of the orbit
  Rem Now rotate from orbital plane to INERTIAL CELESTIAL Coordinates
  X1=X0*C(1,1)+Y0*C(1,2)
  Y1=X0*C(2,1)+Y0*C(2,2)
  Z1=X0*C(3,1)+Y0*C(3,2)
  Rem Rotate through current GHA of Aries, convert to GEOCENTRIC coordinates
  G7=T*G1+G2
  G7=(G7-Int(G7))*P2
  S7=-Sin(G7)
  C7=Cos(G7)
  X=+(X1*C7)-(Y1*S7)
  Y=+(X1*S7)+(Y1*C7)
  Z=Z1
Return
Rem ------------------------------------------------------------
Procedure Fnx
  Rem Routine to extract all the parameters you might ever need
  Rem    First get Vector from observer to satellite--
  X5=(X-X9)
  Y5=(Y-Y9)
  Z5=(Z-Z9)
  R5=Sqr(X5^2+Y5^2+Z5^2)
  Rem Now rotate into observer' s LOCAL Coordinates----
  Rem where X8=North, Y8=East, Z8=Up (Left-handed system)
  Z8=+(X5*C8*C9)+(Y5*S8*C9)+(Z5*S9)
  X8=-(X5*C8*S9)-(Y5*S8*S9)+(Z5*C9)
  Y8=+(Y5*C8)-(X5*S8)
  S5=Z8/R5
  C5=Sqr(1-S5*S5)
  E9=(Atn(S5/C5))/P0                    ! E9=elevation
  Dx=X8
  Dy=Y8
  Gosub Fna
  A9=D/P0                               ! FNA resolves quadrant A9=Azimuth
  Dx=X
  Dy=Y
  Gosub Fna
  W5=360-D/P0                           ! W5=SubSatellitePoint W.long.
  B5=Z/R
  L5=Atn(B5/(Sqr(1-B5^2)))/P0           ! L5=SSP Lat.
Return                               !---note R-R0=sat. alt. above mean spheroid
Rem ------------------------------------------------------------------
Procedure Fna
  Rem Calculates INVERSE TANGENT in proper quadrant ala FORTRAN ATAN2
  If Dx<0
```

```
    D=Pi+Atn(Dy/Dx)         ! Q2 or Q3
  Endif
  If Dx>0                   ! Q1 or Q4
    If Dy>=0
      D=Atn(Dy/Dx)
    Else
      D=P2+Atn(Dy/Dx)
    Endif
  Endif
  Rem The two cases for DX=0
  If Dx=0
    If Dy>=0
      D=Pi/2
    Else
      D=3*Pi/2
    Endif
  Endif
Return
Rem ---------------------------------------------------------------------
Procedure Map
  Cls
  If Mapon!
    Arrayfill Xx%(),-1
    Arrayfill Yy%(),-1
    Restore Map
    For Mj%=0 To 199
      Read Lvt%(Mj%)
    Next Mj%
    Green!=False
    For My%=0 To 199
      Ox%=0
      Nx%=640
      Lvs%=Lvt%(My%)
      Repeat
        If Lvs%<>0
          Read Nx%
        Endif
        If Green!
          Color 3
        Else
          Color 0
        Endif
        Draw Ox%,My% To Nx%,My%
        Ox%=Nx%
        If Lvs%<>0
          Green!=Not Green!
        Endif
        Dec Lvs%
      Until Lvs%<=0
      If Green!
        Color 3
        Draw Nx%,My% To 640,My%
      Endif
```

```
    Next My%
  Endif
  Yy%(0)=1.11111111*(90-L9)
  Wx5=W9+180
  If Wx5>=360
    Wx5=Wx5-360
  Endif
  Xx%(0)=1.77777777*(360-Wx5)
  Color 2
  Circle Xx%(0),Yy%(0),3
  Color 1
  Plot Xx%(0),Yy%(0)
Return
Rem World map data (by John Logajan) in scanned form (for data compression),
Rem left to right, top to bottom (0-639, 0-199.)
Rem The first 200 datums represent the number of color transitions per each
Rem line.  The remaining datums represent each X coordinate at which the
Rem transitions occur -- it is up to the software to keep track of the Y
Rem coordinate based on the transitions-count-per-line data.
Map:
Data 0,0,0,0,0,0,0,2,4,4,8,12,12,18,18,22,16,10,24,20,21,20,19,18,13,10,12,14
Data 16,17,14,12,12,16,18,18,18,16,18,20,18,20,18,10,12,8,10,12,12,16,20,18,20
Data 18,14,20,22,16,16,16,16,14,8,10,10,10,6,12,14,12,14,12,20,14,14,20,18,18
Data 18,18,14,12,14,12,12,12,18,18,16,20,16,12,14,16,12,10,10,10,14,16,18,18,12
Data 16,20,14,18,14,10,10,14,10,10,10,12,14,12,12,13,11,10,8,12,12,10,8,8,8,8,6
Data 6,6,6,6,6,6,8,8,10,6,6,6,6,6,4,6,6,6,8,6,4,4,4,2,4,4,2,4,4,2,4,4,0,0,0,0,0
Data 2,2,2,2,2,4,8,6,6,8,8,4,6,6,6,6,6,4,5,2,0,0,0,0,0,0,0,0,0,0,0,0,0,236,283
Data 163,201,207,285,159,196,203,300,157,182,198,297,352,369,485,495,146,156
Data 164,177,189,194,200,290,339,369,483,498,145,159,162,185,191,288,340,360
Data 490,496,501,507,114,117,121,126,135,139,146,159,161,186,190,287,342,354
Data 356,365,497,509,100,106,112,117,148,158,160,182,193,288,346,351,358,362
Data 436,443,496,520,99,106,113,116,125,131,136,139,141,144,149,156,159,180,196
Data 286,423,443,479,523,567,575,135,138,148,154,175,179,216,287,418,432,473
Data 523,565,578,581,589,98,111,161,171,218,286,416,427,470,522,99,110,114,121
Data 126,132,137,142,146,153,159,178,180,185,220,284,414,423,443,448,462,549
Data 569,576,97,110,111,135,137,141,145,154,160,189,220,282,411,419,442,450,452
Data 551,567,587,0,5,41,45,98,109,110,136,145,155,160,195,220,282,411,421,439
Data 551,554,591,637,3,31,61,88,93,114,139,146,154,170,199,222,282,357,376,414
Data 422,438,605,638,0,30,75,79,107,146,158,165,169,182,201,222,281,350,382,425
Data 434,438,632,0,4,24,121,128,138,142,171,184,202,226,276,345,388,396,402,406
Data 636,0,9,24,122,126,174,182,206,224,267,342,393,397,15,29,175,183,211,224
Data 261,343,394,397,19,22,166,186,210,224,259,277,282,284,294,342,16,21,164
Data 169,174,180,208,225,250,276,296,340,360,363,0,8,13,24,162,166,175,180,205
Data 227,249,277,295,337,359,360,0,14,19,27,159,165,175,191,206,228,248,279,291
Data 333,357,358,639,25,159,181,191,194,206,229,245,306,309,329,352,357,636,24
Data 155,181,193,199,203,231,246,328,351,357,631,24,153,181,197,233,245,328,352
Data 357,597,604,627,28,56,61,152,182,197,203,207,239,244,329,354,360,596,605
Data 617,32,52,73,155,180,199,201,209,311,315,329,353,359,585,588,596,603,613
Data 37,50,75,156,181,211,307,317,330,336,340,350,352,355,358,571,599,611,36,43
Data 47,50,78,161,183,212,307,317,341,350,352,354,357,569,597,610,32,40,45,47
Data 79,165,183,212,309,316,341,349,357,566,596,611,27,37,82,174,179,215,305
Data 307,310,318,334,340,343,346,356,566,596,608,26,29,83,86,87,174,178,219,302
Data 308,310,320,335,342,347,569,572,574,596,609,20,25,83,86,87,176,179,221,302
```

```
Data 308,312,321,329,570,571,575,596,605,84,87,90,143,147,177,179,221,303,308
Data 311,324,328,570,571,575,597,602,626,629,5,7,86,88,92,177,179,222,301,308
Data 310,323,326,570,571,575,598,602,91,216,312,321,323,570,572,576,597,600,92
Data 214,217,222,310,314,321,570,572,576,595,597,95,211,216,221,318,570,572,577
Data 98,161,166,206,214,224,315,570,572,574,99,159,163,165,168,207,216,223,314
Data 568,572,575,99,158,161,214,316,410,414,566,572,575,586,588,99,167,173,213
Data 318,374,378,406,412,426,428,565,571,573,582,587,99,164,168,172,175,210,317
Data 373,379,383,386,404,409,424,428,563,571,576,581,583,99,163,166,173,178,197
Data 202,205,305,345,347,372,389,405,411,561,571,580,98,163,166,174,177,195,303
Data 327,329,332,338,345,350,370,391,405,411,559,568,579,98,196,304,326,335,337
Data 339,349,352,370,377,385,393,407,412,553,568,575,99,193,304,324,334,338,342
Data 352,354,409,413,551,568,572,99,189,304,321,325,327,334,338,346,353,354,363
Data 366,410,413,535,536,549,568,573,100,188,303,320,324,326,334,338,348,351
Data 355,362,366,408,414,533,535,539,542,548,568,573,100,187,303,321,341,350
Data 356,364,366,407,415,531,541,549,567,572,102,186,304,319,342,348,357,363
Data 368,410,415,538,542,550,563,571,103,186,308,317,320,339,345,348,358,362
Data 369,538,544,551,561,571,104,186,310,313,318,340,361,367,378,382,383,534
Data 544,551,556,571,105,185,308,340,363,367,377,381,383,534,544,549,552,569
Data 108,182,305,340,382,535,551,563,111,180,303,346,356,362,381,535,550,559
Data 112,177,303,348,355,366,381,537,551,555,113,117,118,176,302,353,355,537
Data 551,554,113,117,119,176,302,538,114,118,120,162,168,177,295,297,301,406
Data 410,537,116,120,121,151,172,177,290,293,294,296,300,407,410,537,115,121
Data 123,148,172,178,298,382,383,408,411,535,117,122,124,148,173,178,180,183
Data 294,381,384,410,414,534,120,123,125,148,174,178,294,382,385,412,419,533
Data 120,124,127,147,175,177,180,182,293,383,386,412,416,421,428,436,438,532
Data 534,537,122,125,128,147,181,183,292,384,386,425,439,531,533,537,124,126
Data 130,147,171,179,291,384,388,426,442,529,533,536,35,37,39,40,132,147,169
Data 173,176,183,290,386,389,427,442,482,483,524,534,535,38,40,41,43,132,148
Data 159,166,289,387,389,426,443,478,483,513,514,519,41,45,132,149,159,165,184
Data 189,290,387,390,425,449,475,484,510,514,518,42,45,132,149,158,165,185,194
Data 290,387,392,423,449,473,486,509,513,518,135,165,187,191,197,201,291,388
Data 393,423,449,471,486,509,513,517,534,538,138,164,198,201,291,389,394,421
Data 450,469,488,510,533,538,141,164,290,390,395,418,450,467,487,512,533,538
Data 145,171,290,391,395,414,450,465,487,491,493,513,532,537,155,173,289,394
Data 396,412,451,463,493,514,533,537,156,173,290,395,396,407,452,463,494,515
Data 534,541,160,172,290,404,452,463,495,515,534,537,538,541,164,172,191,198
Data 290,398,414,418,453,463,495,498,501,515,534,537,540,544,166,172,187,199
Data 292,398,404,412,454,462,495,498,502,515,532,533,536,544,167,173,185,212
Data 293,411,455,462,495,497,504,512,531,533,536,543,167,174,177,181,184,212
Data 295,411,455,461,462,464,495,498,506,510,529,532,537,545,171,214,296,410
Data 456,460,461,465,494,499,506,508,528,530,538,545,174,178,180,216,297,409
Data 461,466,494,499,537,546,181,218,298,409,461,466,495,501,527,530,536,538
Data 540,546,182,225,301,324,328,408,462,465,489,491,496,503,526,531,540,545
Data 182,228,303,319,329,407,489,495,498,504,523,533,182,230,330,405,490,497
Data 499,504,521,531,182,230,335,404,492,498,500,504,518,529,180,230,337,403
Data 493,500,517,530,546,548,178,232,336,400,492,504,515,532,534,538,541,543
Data 546,549,157,158,177,231,336,398,496,505,515,532,533,535,537,542,546,549
Data 157,159,176,237,335,376,381,397,497,505,515,529,532,534,546,548,551,562
Data 157,158,176,241,336,375,380,395,498,509,515,529,532,536,546,549,552,567
Data 176,244,336,394,499,510,515,528,531,535,554,571,177,251,337,392,500,510
Data 516,527,531,541,544,553,554,576,589,591,175,253,339,391,502,509,523,526
Data 532,535,536,539,544,547,551,553,556,578,589,593,175,258,340,390,503,509
Data 532,535,537,540,562,580,588,592,175,258,341,390,505,510,532,535,538,540
```

```
Data 558,560,565,582,585,591,595,597,176,259,342,391,512,523,558,560,566,582
Data 586,588,595,598,178,259,342,390,509,525,543,546,566,584,179,259,343,391
Data 517,547,569,576,580,584,180,258,343,391,531,535,539,544,570,575,581,587
Data 603,606,180,257,343,393,539,542,582,589,605,609,181,255,344,393,551,557
Data 572,574,608,609,182,254,344,393,406,409,551,564,572,575,12,13,184,253,342
Data 392,406,409,551,564,572,576,12,16,184,251,342,393,404,410,544,547,550,562
Data 571,576,185,251,341,393,404,410,542,562,571,579,616,618,186,251,340,392
Data 400,410,541,565,571,579,616,618,189,251,340,391,398,409,537,568,570,580
Data 617,619,637,0,192,251,340,390,398,408,537,580,635,638,195,250,341,386,398
Data 408,536,581,635,638,195,250,342,384,398,408,535,583,195,249,342,382,398
Data 407,421,423,530,585,612,613,195,248,343,383,397,407,418,420,525,586,612
Data 616,195,248,344,383,396,406,522,588,614,618,194,246,345,384,396,405,522
Data 589,194,240,345,384,397,405,521,590,194,237,345,383,397,405,521,592,194
Data 235,346,381,399,404,521,592,194,234,346,379,521,593,193,234,347,379,522
Data 593,193,234,348,378,523,594,192,233,349,377,524,593,192,231,350,375,524
Data 593,192,230,350,375,525,593,192,230,352,373,525,592,192,227,351,372,525
Data 543,558,591,192,226,352,369,524,541,559,590,191,225,354,365,524,533,560
Data 589,627,629,191,219,562,588,627,631,190,220,563,588,628,633,189,219,568
Data 587,630,638,188,218,570,586,630,637,189,215,574,581,628,637,188,210,629
Data 635,188,210,577,584,626,634,188,207,577,584,624,633,188,207,578,584,623
Data 630,5,7,187,205,579,582,620,628,6,7,187,205,617,626,187,204,616,625,185
Data 201,616,624,185,204,617,620,185,204,185,201,442,446,185,200,442,445,186
Data 198,186,198,213,218,186,199,214,216,188,200,189,205,252,257,193,205,255
Data 257,237,241,215,218,212,218,212,220,207,219,205,216,545,551,203,215,497
Data 511,516,528,544,566,196,214,409,425,466,577,196,211,394,444,460,582,192
Data 210,379,446,456,587,593,601,185,189,192,210,332,344,369,607,184,211,306
Data 618,173,178,184,212,298,625,137,158,171,212,295,624,100,113,127,213,287
Data 620,97,114,125,213,279,617,70,88,95,211,275,613,54,209,271,610,0,219,262
Data 610,627,226,256
Rem -- SYMBOL TABLE -- (ala Clark and Tilley)
Rem A0 = SEMI-MAJOR axis
Rem A9 = Azimuth
Rem Cn = COSines, C(J,K) also
Rem E0 = eccentricity of orbit
Rem E8 = observer horizion
Rem E9 = satellite elevation
Rem G0 = GM of Earth
Rem G1 = Sidereal/Solar
Rem G2 = New Year's sidereal time
Rem G7 = GHA Aries
Rem H1 = Hours duration
Rem H3 = Hour epoch
Rem H4 = Hour
Rem H9 = observer height
Rem K0 = initial orbit #
Rem K  = orbit counter
Rem K7 = INT(T)
Rem L5 = SSP LATITUDE
Rem L8 = GEOCENTRIC latitude
Rem L9 = observer longitude
Rem M  = mean anomaly
Rem M0 = mean anomaly of ref epoch
Rem M3 = ref epoch minutes
```

```
Rem M4 = integer minutes
Rem M5 = difference in Transcendental Eq.
Rem N0 = mean motion
Rem O, O0 = R.A.A.N.
Rem P0,P1,P2 variations on a theme of PI
Rem Q,Q0 = total orbits
Rem R = ? in plane of orbit
Rem R5 = range
Rem R6 = previous range
Rem R9 intermediate for GEOCENTRICs
Rem Ss used for SIN
Rem T0 = ref time in orbit days
Rem W, W0 arg of perigee
Rem W5 = SSP W. long.
Rem W9 = observers W. long
Rem X0,Y0,R coord in plane of orbit
Rem X1,Y1,Z1 "    " INERTIAL CELESTIAL
Rem X,Y,Z in GEOCENTRIC coordinates
Rem X9,Y9,R9 are observer's GEOCENTRIC coordinates
Rem X5,Y5,Z5 ==> R5  vector from observer to satellite
Rem X8,Y8,Z8 are observer's LOCAL Coordinates

SHAR_EOF
cat << \SHAR_EOF > orbitals.dat

The most current orbital elements from the NASA Prediction Bulletins are
carried on the Celestial RCP/M, (513) 427-0674, and are updated several times
weekly.  Documentation and tracking software are also available on this
system.

The Celestial RCP/M may be accessed 24 hours/day at 300, 1200, or 2400 baud
using 8 data bits, 1 stop bit, no parity.

- Current NASA Prediction Bulletins #546 -
Alouette 1
1 00424U          89206.16793576 0.00000172           19720-3 0  2241
2 00424  80.4633 248.4238 0024150  36.2692 324.0085 13.67172480338002
SOLRAD R/B
1 00727U 64001  A 89198.56177170  .00000164  00000-0  15604-3 0  2797
2 00727  69.9022  30.4841 0011954  55.4275 304.7932 13.93553262297324
Cos 185 R/B
1 03019U 67104  B 89198.54646773  .00004760  00000-0  32780-3 0  3891
2 03019  64.0624  18.9846 0219843 225.3173 132.9859 14.89572043109609
ATS 3
1 03029U 67111  A 89187.56917568 -.00000074  00000-0  99999-4 0  2365
2 03029  12.7757  24.1274 0015540 194.3702 165.6963  1.00272176 79330
Cosmos 398
1 04966U 71 16  A 89199.72951283 -.00016207  00000-0 -10554-3 0  8303
2 04966  51.5658 319.5495 2511049 266.1090  65.2085 10.54262717555713
LAGEOS
1 08820U          89206.77237598 0.00000005                   0  8285
2 08820 109.8345 243.2992 0044897 307.8156  51.8574  6.38664132 53099
GOES 2
```

```
1 10061U          89193.69270730 -.00000018                   0  2869
2 10061   7.2367  67.9134 0005870  97.5785 262.2768  1.00279674  5639
GPS-0001
1 10684U          89202.85871690 0.00000011           10000-3 0  1698
2 10684  63.6023  99.7113 0108902 198.3317 161.3557  2.00559884 69221
GPS-0002
1 10893U          89201.32278087 -.00000027                   0    87
2 10893  64.4770 340.6383 0154404  28.1763 332.6305  2.00549743 82010
GOES 3
1 10953U          89198.73487946 0.00000088           10000-3 0  6896
2 10953   6.1136  70.5293 0006800 259.6167 100.2746  1.00274815  1248
SeaSat 1
1 10967U          89197.23438422 0.00000977           38799-3 0  1437
2 10967 108.0115 347.4202 0002168 235.2265 124.8673 14.34718776578251
GPS-0003
1 11054U          89204.62709625 -.00000027                   0   653
2 11054  64.0167 336.9646 0060222 114.0010 246.6680  2.00561675 79092
GPS-0004
1 11141U          89204.72662801 0.00000010                   0  1804
2 11141  63.5822  99.6011 0057467 321.4893  38.1454  2.00571030 77794
NOAA 6
1 11416U          89200.05544746 0.00000829           36445-3 0  8753
2 11416  98.5094 196.1955 0012703  22.3345 337.8387 14.25872183522445
Solar Max
1 11703U          89201.89658051 0.00059458           78802-3 0   285
2 11703  28.4937  40.2847 0000586  68.4626 291.6007 15.56850600525095
GPS-0006
1 11783U          89202.68997067 -.00000027                   0  9317
2 11783  63.9528 336.6909 0143336  61.9063 299.5654  2.00568321 67691
GOES 4
1 11964U          89198.09983592 -.00000043                   0   594
2 11964   5.5501  73.3770 0033168  94.9661 265.3246  0.99228840  2329
GOES 5
1 12472U          89191.00000000 -.00000249           10000-3 0  7407
2 12472   2.6193  79.9038 0001682 120.2420  23.8683  1.00267489 28808
SME
1 12887U 81100  A 89198.60724050  .00015696  00000-0  53802-3 0  2468
2 12887  97.6841 225.6881 0003215 118.0081 242.1476 15.30277523429809
UOSAT 1
1 12888U          89204.08152074 0.00099250           89255-3 0  6439
2 12888  97.5502 260.2192 0000794 345.5117  14.5401 15.67921102434601
Meteor 2-08
1 13113U          89204.00241408 0.00000167           14110-3 0  6564
2 13113  82.5370  30.2903 0015066 327.4850  32.5383 13.84020055370192
Salyut 7
1 13138U          89206.08301020 0.00004463           10793-3 0  6700
2 13138  51.6084 131.6728 0001507 128.3129 231.8984 15.43251658414047
LandSat 4
1 13367U 82 72  A 89198.64210169  .00001099  00000-0  25410-3 0   257
2 13367  98.2119 261.8410 0003570  97.4120 262.7608 14.57119054372585
Meteor 2-09
1 13718U          89198.61733429 0.00000206           10247-3 0  7970
2 13718  81.2447 251.7785 0054867 272.8515  86.6395 14.13238180339859
```

```
IRAS
1 13777U          89199.57374005 -.00000334          -22041-3 0  6598
2 13777  99.0426  36.2509 0012030 223.9617 136.0646 13.98617482   306
TDRS 1
1 13969U 83 26  B 89193.61106535 -.00000183  00000-0  00000 0 0  8197
2 13969   3.6420  68.4417 0002893 137.6679 225.5675  1.00123858 89665
GOES 6
1 14050U          89197.78372817 0.00000114                   0  9878
2 14050   1.3661  82.3935 0001858 111.0113 248.3075  1.00287640  6901
OSCAR 10
1 14129U          89202.10971599 -.00000084                   0  4175
2 14129  26.0887 253.9637 6053670  60.6568 346.2092  2.05883104 17918
GPS-0008
1 14189U          89204.77202103 0.00000010           10000-3 0  6472
2 14189  63.2041  98.2080 0136478 215.3731 143.7755  2.00567719 44174
Meteor 2-10
1 14452U          89204.06734195 0.00000522           21975-3 0  7515
2 14452  81.1648 257.5931 0095433 354.0213   5.9821 14.22260082297640
LandSat 5
1 14780U          89207.04971353 0.00000739           17364-3 0  8416
2 14780  98.1656 268.5478 0003865 111.5435 248.6087 14.57132172287228
UOSAT 2
1 14781U 84 21  B 89199.62625717  .00001626  00000-0  32042-3 0  4813
2 14781  98.0027 257.5120 0012166 219.4269 140.6050 14.63764368287146
LDEF
1 14898U          89203.36779891 0.00028572           43127-3 0  9142
2 14898  28.5060 288.4742 0000305 293.9869  66.0580 15.53295791296922
GPS-0009
1 15039U          89203.83458933 0.00000009                   0  7055
2 15039  62.7617  97.2295 0012949 206.4440 153.6685  2.00568700 37437
Meteor 2-11
1 15099U          89198.39046606 0.00000145           12119-3 0  9573
2 15099  82.5295 342.3507 0013080 147.4673 212.7293 13.83679414254299
GPS-0010
1 15271U          89205.72943660 -.00000027                   0  6756
2 15271  63.3556 336.3361 0100019 321.0446  38.0588  2.00566524 34584
Cosmos 1602
1 15331U          89194.80061499 0.00002232           32106-3 0   546
2 15331  82.5390 313.8830 0025148   1.4155 358.7130 14.75782144258144
NOAA 9
1 15427U          89196.47764042 0.00000488           28734-3 0  4043
2 15427  99.1448 183.9754 0014425 233.1046 126.8803 14.12086380236448
Meteor 2-12
1 15516U          89204.60370140 0.00000358           31246-3 0  1000
2 15516  82.5350 275.7939 0017505  17.9536 342.2241 13.84145796226099
Cosmos 1686
1 16095U          89205.69444412 0.00020490           46125-3 0  3190
2 16095  51.6090 133.5796 0001120 115.5547 244.4815 15.43260670215088
GPS-0011
1 16129U          89205.88500538 0.00000010                   0  3410
2 16129  63.7328  98.1204 0118094 149.8658 210.8873  2.00563934 27794
Meteor 3-1
1 16191U          89204.98012498 0.00000043           10000-3 0  8418
```

2 16191 82.5457 209.7207 0018896 204.8083 155.2143 13.16870606180362
Meteor 2-13
1 16408U   89204.97574168 0.00000112   94947-4 0 5153
2 16408 82.5350 190.3584 0015447 192.8233 167.2520 13.84191299180642
Mir
1 16609U   89205.39540184 -.00008640   -11215-3 0 9543
2 16609 51.6196 177.4198 0009423 191.1241 168.9523 15.57421363196968
SPOT 1
1 16613U   89207.05108603 0.00000626   31158-3 0 5390
2 16613 98.7367 281.0995 0001802 81.9325 278.2091 14.20034337 17646
Meteor 2-14
1 16735U   89205.02762104 0.00000109   93312-4 0 3094
2 16735 82.5350 217.1531 0012761 270.4679 89.5000 13.83938849159552
Cosmos 1766
1 16881U   89207.05339059 0.00001437   20739-3 0 6775
2 16881 82.5248   2.0576 0022790 337.5195 22.5000 14.75440085161025
EGP
1 16908U   89202.18032835 -.00000038   77810-5 0 1479
2 16908 50.0064 193.5888 0011257 74.2487 285.9588 12.44379520133640
FO-12
1 16909U   89197.10014353 -.00000025   10000-3 0 1630
2 16909 50.0167 209.4498 0011078 60.4152 299.7766 12.44400597133006
NOAA 10
1 16969U   89200.21509624 0.00000598   28162-3 0 2513
2 16969 98.6366 230.0955 0013415 153.2927 206.8943 14.23090983148495
Meteor 2-15
1 17290U   89204.23805203 0.00000216   18513-3 0 2728
2 17290 82.4688 124.8780 0012784 154.3441 205.8358 13.83741427128609
MOS-1
1 17527U 87 18 A 89199.02136141 -.00000388 00000-0 -27596-3 0 1872
2 17527 99.1484 270.2132 0000531 56.6570 303.4662 13.94831938122675
GOES 7
1 17561U 87 22 A 89196.48328078 -.00000046 00000-0 99999-4 0 3008
2 17561   0.0171 117.6897 0002124 328.5324 273.7952 1.00272540 2048
Kvant
1 17845U   89205.58789710 0.00027395   37108-3 0 8192
2 17845 51.6217 176.4560 0008890 193.4524 166.6002 15.57436073133268
DMSP B5D2-3
1 18123U 87 53 A 89197.02695732 .00000640 00000-0 35987-3 0 3478
2 18123 98.8252 27.1514 0013117 260.6515 99.3174 14.13471492106904
RS-10/11
1 18129U   89206.02614146 0.00000078   79289-4 0 8262
2 18129 82.9239 210.1998 0010652 312.2194 47.8085 13.71991825104588
Cosmos 1867
1 18187U 87 60 A 89198.97796186 .00000132 00000-0 69311-4 0 8858
2 18187 65.0147 296.5794 0019568 256.3044 103.5840 14.29382746105525
Cosmos 1870
1 18225U   89206.76974459 0.00141952 11453-4 16950-3 0 1552
2 18225 71.8628 310.9157 0016331 252.2513 107.7208 16.07994556117480
Meteor 2-16
1 18312U   89204.42835747 0.00000219   18868-3 0 2952
2 18312 82.5524 190.8171 0013701 90.0786 270.1941 13.83494008 97529
Meteor 2-17

```
1 18820U          89205.69077344 0.00000176           14776-3 0  1356
2 18820  82.5427 250.7488 0016955 159.4792 200.7026 13.84186077 74874
DMSP B5D2-4
1 18822U 88  6  A 89198.42622926  .00000551  00000-0  27225-3 0  2468
2 18822  98.6986  78.6651 0006110 152.8227 207.3284 14.20830029 75295
AO-13
1 19216U          89186.15508008 -.00000132           10000-3 0   382
2 19216  57.2070 200.8551 6748913 207.0433  90.2906  2.09703317  8118
OKEAN 1
1 19274U 88 56  A 89199.05309436  .00002434  00000-0  35763-3 0  4351
2 19274  82.5184 108.0916 0024166 128.8964 231.4413 14.74959851 55635
Meteor 3-2
1 19336U          89204.81473383 0.00000391           10000-2 0  2226
2 19336  82.5418 149.7562 0019050  54.2703 306.0096 13.16855216 47735
NOAA 11
1 19531U          89199.89876582 0.00000494           29490-3 0  1061
2 19531  98.9414 143.7651 0012043 138.5761 221.6325 14.11072946 41948
TDRS 3
1 19548U 88 91  B 89153.77764324  .00000129  00000-0  99999-4 0   261
2 19548   0.5410  85.9584 0002050  10.4654 263.5580  1.00263057  1649
GPS-0014
1 19802U 89 13  A 89195.92402568  .00000015  00000-0  99999-4 0   489
2 19802  55.1018 210.7161 0066687 157.4981 202.8516  2.00556819  2987
Meteor 2-18
1 19851U          89204.99868031 0.00000117           10000-3 0   589
2 19851  82.5237 129.9222 0013482 200.0593 160.0041 13.83820641 20175
TDRS 4
1 19883U 89 21  B 89193.30721528 -.00000234  00000-0  99999-4 0   201
2 19883   0.4184  71.6153 0001726  52.2881 235.9737  1.00264167   477
Cosmos 2022
1 20024U          89180.39981994 -.00000011           10000-2 0   363
2 20024  64.8272  50.8719 0001867 255.3129 104.7148  2.13102677  6110
Cosmos 2023
1 20025U          89180.69329080 -.00000011           10000-3 0   337
2 20025  64.8449  50.8407 0013705 210.6426 149.3398  2.13102719   639
Cosmos 2024
1 20026U          89180.38648897 -.00000011           10000-3 0   312
2 20026  64.8327  50.8698 0008813 267.3366  92.6243  2.13204691   623
SuperBird A
1 20040U          89183.78104506 -.00000099           10000-3 0   180
2 20040   0.0262 313.9524 0003825 147.9338 258.1480  1.00273104   239
1989 041C
1 20042U          89185.08517834 0.00030059           39919-2 0   287
2 20042   6.9331  37.4445 7284280 200.3317 100.7896  2.29417718   558
Cosmos 2026
1 20045U          89182.87143968 0.00000147           14176-3 0   337
2 20045  82.9397 343.5786 0038435 224.9740 134.8304 13.75019694  3391
1989 042B
1 20046U          89190.25933410 -.00005554          -56739-2 0   517
2 20046  82.9417 338.0442 0035026 199.2664 160.7174 13.76469147  4415
Molniya3-35
1 20052U          89189.80933893 0.00000326           -81454-3 0   444
2 20052  62.9521  99.8751 7379264 288.3629   9.3770  2.00620985   616
```

```
1989 043D
1 20055U          89187.82942194 0.00000550           19192-2 0   268
2 20055  62.8484 100.2186 7414419 288.3220   9.1933  1.95983490   552
GPS-0013
1 20061U          89195.11831124 -.00000029                  0   285
2 20061  54.5868  29.5793 0080698 163.2427 197.0849  2.00573534   713
1989 044B
1 20062U          89193.27933477 0.18331712 -23867-4  49193-3 0   674
2 20062  30.2951 166.4880 0064540   2.3773 357.7413 16.35924744  3699
1989 044C
1 20063U          89190.62410034 0.00111825  17032-4  12205-2 0   390
2 20063  37.5403   7.2226 6012784 224.5458  67.5657  4.12779176  1205
Cosmos 2027
1 20064U          89180.22022443 0.00008604           36818-3 0   107
2 20064  65.8385  25.6376 0024329 340.4779  20.7342 15.23081660  2240
1989 045B
1 20065U          89189.58668525 0.00012737           51370-3 0   330
2 20065  65.8410 356.1109 0028402 352.2314   7.8374 15.24711048  3678
Raduga 1-1
1 20083U          89206.78146626 0.00000150           10000-3 0   381
2 20083   1.4049 273.2133 0005585 348.2342  12.0835  1.00271194   344
1989 048D
1 20086U          89194.20510965 -.00000025           10000-3 0    76
2 20086   1.4319 273.6399 0021177 326.9784  33.7202  0.97889488   224
Nadezhda
1 20103U          89207.11643209 0.00000084           83861-4 0   170
2 20103  82.9622 165.3359 0036332 225.3657 134.4533 13.73490787  2954
1989 050B
1 20104U          89203.09208317 0.00000006                  0   107
2 20104  82.9619 168.2739 0032035 214.1965 145.7130 13.74883091  2390
Gorizont 18
1 20107U          89206.53673588 -.00000252           10000-3 0   219
2 20107   1.4679 275.1362 0002553  36.6151 324.4230  1.00278437   204
1989 052D
1 20110U          89198.38268624 -.00000040           10000-3 0    65
2 20110   1.5129 275.7936 0028263  76.8720 284.4159  1.03064188   121
Olympus
1 20122U          89206.45833329 -.00000133           10000-3 0   137
2 20122   0.3111 116.5200 0039730 161.1000 186.6730  0.99668381    02
1989 053B
1 20123U          89202.09859699 0.00114392           97589-2 0   130
2 20123   6.1734 103.5478 7316792 185.4997 162.2780  2.26199326   227
Cosmos 2030
1 20124U          89207.14165407 0.01706290  29258-4  32748-3 0   402
2 20124  67.1451  96.1609 0067734  59.2471 301.7086 16.24824071  2182
1989 055A
1 20134U          89206.75123345 0.00119290  52645-5  19047-3 0   225
2 20134  82.5685 101.3308 0010288 271.6879  88.3357 16.03507280  1187
1989 056A
1 20136U          89206.77059950 0.00212921  26538-4  38047-3 0   215
2 20136  50.5522  85.5202 0092144  44.9697 315.8559 15.93565154  1173
1989 057A
1 20145U          89206.75002318 0.00051150 -19215-7  10033-3 0   239
```

```
2 20145  82.3116  93.3835 0097667  75.8591 285.3595 15.91659320   868
1989 058A
1 20147U          89206.79664603 -.00025164          -41389-3 0    99
2 20147  65.0243 330.8234 0009661 269.1416  90.7071 15.52051461   284
1989 059A
1 20149U          89207.11359027 0.00018128           19706-1 0    73
2 20149  82.9394  87.6575 0030268 276.4523  83.3026 13.72429037   111
1989 059B
1 20150U          89207.03988152 0.00034193           36088-1 0    42
2 20150  82.9381  87.7138 0023133 269.7533  90.1012 13.74127805    92
--
Dr TS Kelso                           Asst Professor of Space Operations
tkelso@blackbird.afit.af.mil          Air Force Institute of Technology

SHAR_EOF
#       End of shell archive
exit 0


From 4859356593891757922
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,174614567da5ed15,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-08-07 16:11:18 PST
Path: utzoo!utgpu!watmath!att!tut.cis.ohio-state.edu!mailrus!csd4.milw.wisc.edu!
uakari.primate.wisc.edu!indri!ames!sun-barr!newstop!sun!sunfedcomm!grapevine!
koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i067: diff -- File difference utility
Keywords: C
Message-ID: <34052@grapevine.uucp>
Date: 7 Aug 89 23:11:18 GMT
Sender: news@grapevine.uucp
Lines: 254
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: Schutte@nikhefh.hep.nl (Klamer Schutte)
Posting-number: Volume 2, Issue 67
Archive-name: diff

Here is the source of a diff(1) program. I wrote it some time ago, but
decided to submit now because of questions of such in comp.sys.atari.st.

There is no other copyright on the program than mentioned in its header --
I am the writer myself.

Compile, enjoy!
Klamer.

--
------------------------------------------------------------------------------
Klamer Schutte      mcvax!nikhefh!{n62,Schutte}     {Schutte,n62}@nikhefh.hep.nl
```

```
------------ cut here -------------

/*
 *   diff.c
 *   program to compare text files
 *   prelimary release: - only limited testing performed.
 *                      - no context command as in (BSD) U*IX.
 *   Compiled using megamax. Other compilers shouldn't present problems.
 *   Permission granted for using this source if:
 *        - Credit to my name is made in the resulting program.
 *        - No profit is made on the program.
 *          When you want to use this in a commercial product
 *          please contact me.
 *
 *   Klamer Schutte
 *   Tolstraat 46-3
 *   1073 SC  Amsterdam
 *   The Netherlands
 *   tel: 31 20 714578
 *   Schutte@nikhefh.hep.nl   \
 *   schutte@thoff.uucp       - for the time being
 *   mcvax!nikhefh!Schutte    /
 */

#include <stdio.h>
#include <errno.h>

#define   MAXLINE   75   /* screen width - space for '>nnn ' */

struct line
{    char data[MAXLINE];
     struct line   *next;
     int  line_nr;
}

main( argc, argv )
int  argc;
char *argv[];
{    FILE *fp1, *fp2;
     struct line   *top1 = NULL, *top2 = NULL,
                   *cur1, *cur2, *last1, *last2;
     int  linenr1 = 0, linenr2 = 0, inc1 = 1, inc2 = 1,
          eof = 0,  /* flag where eof is detected */
          len;      /* used in test on '\n' */
/* #[   init: check argc, open fp1, fp2 */
     if (argc != 3)
     {    fputs( "Usage: diff file1 file2" , stderr );
          exit( -1 );
     }
     if (fp1 = fopen( argv[1], "r" ), fp1 == NULL)
     {    fprintf( stderr, "Error opening %s\n", argv[1] );
          exit( errno );
```

```
     }
     if (fp2 = fopen( argv[2], "r" ), fp2 == NULL)
     {    fprintf( stderr, "Error opening %s\n", argv[2] );
          exit( errno );
     }
/* #]   init: */
     while( 1 )
/* #[    main loop: exit: function end on eof other file */
     {
          if (eof != 1)
          {
               cur1 = (struct line * ) malloc( sizeof( struct line ));
cur1->next = NULL;
               if (cur1 == NULL)
               {    fprintf( stderr, "diff: out of memory\n");
                    exit( -39 );
               }
               if (fgets( cur1->data, MAXLINE, fp1 ) == NULL)
                  {
                    if (eof == 0)
                    {    eof = 1;
                         continue;
                    }
                    if (top1 != NULL)
                         end1( top1, "< ");
                    if (top2 != NULL)
                         end1( top2, "> ");
                    exit( 0 );
               }
               linenr1 += inc1;
                  if (cur1->data[ len = (strlen(cur1->data) - 1)] == '\n')
                  {cur1->data[ len ] = 0;
                    inc1 = 1;
                  } else
                    inc1 = 0;
               for( cur2 = top2; cur2 != NULL; cur2 = cur2->next)
/* #[                 check on match: if cur1 == NULL match */
                  {
                    if (strcmp( cur2->data, cur1->data ) == 0)
                    {
                         pr_all( top1, last1, "< " );
                         free( cur1 );
                         cur1 = NULL;
                         top1 = NULL;
                         last1 = NULL;
                         pr_upto( top2, cur2, "> ");
                         if (last2 == cur2)
                         {    last2 = NULL;
                              top2 = NULL;
                         } else
                              top2 = cur2->next;
                         free( cur2 );
                         break;
```

```
                    }
               }
/* #]                  check on match: */
               if (cur1 != NULL)
               {
                    if (last1 != NULL)
                         last1->next = cur1;
                    else
                         top1 = cur1;
                    cur1->line_nr = linenr1;
                    cur1->next == NULL;
                    last1 = cur1;
               }
          }
          /* The linked list from top1 to last1 has cur1 added now
             No record in this list compares to a record in the top2 list
             Read now a line from fp2 and compare this to the top1 list
          */
          if (eof != 2)
          {
               cur2 = (struct line * ) malloc( sizeof( struct line ));
cur2->next = NULL;
               if (cur2 == NULL)
               {    fprintf( stderr, "diff: out of memory\n");
                    exit( -39 );
               }
               if (fgets( cur2->data, MAXLINE, fp2 ) == NULL)
               {
                    if (eof == 0)
                    {    eof = 2;
                         continue;
                    }
                    if (top2 != NULL)
                         end1( top2, "> " );
                    if (top1 != NULL)
                         end1( top1, "< ");
                    exit( 0 );
               }
               linenr2 += inc2;
               if (cur2->data[ len = (strlen(cur2->data) - 1)] == '\n')
                  {cur2->data[ len ] = 0;
                    inc2 = 1;
               } else
                    inc2 = 0;
                  for( cur1 = top1; cur1 != NULL; cur1 = cur1->next )
/* #[                 check on match: if cur2 == NULL match */
                  {
                    if (strcmp( cur2->data, cur1->data ) == 0)
                    {
                         pr_all( top2, last2, "> " );
                         free( cur2 );
                         cur2 = NULL;
                         top2 = NULL;
```

```
                        last2 = NULL;
                        pr_upto( top1, cur1, "< ");
                        if (last1 == cur1)
                        {    last1 = NULL;
                             top1 = NULL;
                        } else
                             top1 = cur1->next;
                        free( cur1 );
                        break;
                   }
              }
/* #]              check on match: */
              if (cur2 != NULL)
              {
                   if( last2 != NULL)
                        last2->next = cur2;
                   else
                        top2 = cur2;
                   cur2->line_nr = linenr2;
                   cur2->next == NULL;
                   last2 = cur2;
              }
         }
     }
/* #]     main loop: */
}

/* #[ end: end1( struct line *, char *) end2( FILE *, struct line *, char *) */
end1( top, prefix )
struct line   *top;
char *prefix;
{    for( ; top != NULL; top = top->next)
          printf( "%3d%s%s\n", top->line_nr, prefix, top->data );
}

end2( fp, linenr, prefix )
FILE *fp;
int  linenr;
char *prefix;
{    int  len; /* used for checking & correcting '\n' on eoln */
     char line[MAXLINE];
     while( fgets( line, MAXLINE, fp) != NULL)
     {    if (line[ len = (strlen(line) - 1)] == '\n')
             {line[ len ] = 0;
              linenr++;
             }
             printf( "%3d%s%s\n", linenr, prefix, line );
     }
     exit( 0 );
}
/* #] end:*/

/* #[ pr_all: pr_all( struct line *, struct line *, char *) */
```

```
pr_all( top, last, prefix )
struct line   *top, *last;
char *prefix;
{    struct line   *cur;
     if (last != NULL)
          last = last->next;
     for( cur = top; cur != last && cur != NULL; cur = top->next, free(top))
     {    printf( "%3d%s%s\n", cur->line_nr, prefix, cur->data );
          top = cur;
     }
     if (cur != last && cur == NULL)
          fputs( "Internal Error in pr_all\n", stderr );
}
/* #] pr_all: */

/* #[ pr_upto: pr_upto( struct line *, struct line*, char * ) */
pr_upto( top, upto, prefix )
struct line   *top, *upto;
char *prefix;
{    struct line   *cur;
     for( cur = top; cur != upto; cur = cur->next, free( top ))
     {    printf( "%3d%s%s\n", cur->line_nr, prefix, cur->data );
          top = cur;
     }
}
/* #] pr_upto:*/
```

```
From -6980406727061111656
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,462896d8a9bebe64,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-08-08 15:12:58 PST
Path: utzoo!utgpu!watmath!att!pacbell!ames!purdue!decwrl!sun-barr!newstop!sun!
grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i068: emacs.diff -- Diffs to GNU-EMACS 18.52 for Atari  part01/02
Keywords: shar, diff
Message-ID: <34058@grapevine.uucp>
Date: 8 Aug 89 22:12:58 GMT
Sender: news@grapevine.uucp
Lines: 1180
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: uunet.UU.NET!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 2, Issue 68
Archive-name: emacs.diff/part01

Here are the context diffs from GNU-Emacs version 18.52 to the corresponding
atari st version. You need to get the original sources of emacs and apply
patch with this file as input.
You may want to wait for my patches to the GCC library before trying to compile
```

this. I am currently adding some comments and will post the library as soon as
possible.
The binary version of GNU Emacs 18.51 (ST) was posted in the binaries group.

- Edgar

[I have split this into two parts. You will need to stick them together
 manually before unsharring. -sg]

```
-------------------------------------------------------------------------------
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of shell archive."
# Contents:  cdiff-18.52-atari s-tos.h m-atari.h
# Wrapped by roeder@helga on Tue Aug  8 19:40:51 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'cdiff-18.52-atari' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'cdiff-18.52-atari'\"
else
echo shar: Extracting \"'cdiff-18.52-atari'\" \(45058 characters\)
sed "s/^X//" >'cdiff-18.52-atari' <<'END_OF_FILE'
XOnly in .: Makefile
Xdiff -c original/alloc.c ./alloc.c
X*** original/alloc.c   Mon Jul 17 20:52:25 1989
X--- ./alloc.c     Mon Jul 17 21:51:21 1989
X***************
X*** 1438,1450 ****
X    DEFVAR_INT ("pure-bytes-used", &pureptr,
X      "Number of bytes of sharable Lisp data allocated so far.");
X
X- #if 0
X    DEFVAR_INT ("data-bytes-used", &malloc_sbrk_used,
X      "Number of bytes of unshared memory allocated in this session.");
X
X    DEFVAR_INT ("data-bytes-free", &malloc_sbrk_unused,
X      "Number of bytes of unshared memory remaining available in this session.");
X- #endif
X
X    DEFVAR_LISP ("purify-flag", &Vpurify_flag,
X      "Non-nil means loading Lisp code in order to dump an executable.");
X--- 1441,1451 ----
Xdiff -c original/callproc.c ./callproc.c
X*** original/callproc.c    Mon Jul 17 20:52:29 1989
X--- ./callproc.c  Mon Jul 17 20:47:45 1989
X***************
X*** 41,46 ****
X--- 41,50 ----
X
X  #define max(a, b) ((a) > (b) ? (a) : (b))
```

```
X
X+ #ifdef atarist
X+ Lisp_Object Vshell_tmpfile;
X+ #endif
X+
X  Lisp_Object Vexec_path, Vexec_directory;
X
X  Lisp_Object Vshell_file_name;
X***************
X*** 63,73 ****
X--- 67,79 ----
X  call_process_cleanup (fdpid)
X       Lisp_Object fdpid;
X  {
X+ #ifndef atarist
X    register Lisp_Object fd, pid;
X    fd = Fcar (fdpid);
X    pid = Fcdr (fdpid);
X    close (XFASTINT (fd));
X    kill (XFASTINT (pid), SIGKILL);
X+ #endif
X    return Qnil;
X  }
X
X***************
X*** 92,98 ****
X--- 98,108 ----
X  {
X    Lisp_Object display, buffer, path;
X    int fd[2];
X+ #ifdef atarist
X+   unsigned char *filefd;
X+ #else
X    int filefd;
X+ #endif
X    register int pid;
X    char buf[1024];
X    int count = specpdl_ptr - specpdl;
X***************
X*** 140,145 ****
X--- 150,158 ----
X      new_argv[i - 3] = 0;
X    }
X
X+ #ifdef atarist
X+   filefd = XSTRING (args[1])->data;
X+ #else
X    filefd = open (XSTRING (args[1])->data, O_RDONLY, 0);
X    if (filefd < 0)
X      {
X***************
X*** 153,158 ****
X--- 166,172 ----
```

```
X        report_file_error ("Searching for program", Fcons (args[0], Qnil));
X      }
X    new_argv[0] = XSTRING (path)->data;
X+ #endif
X
X    if (XTYPE (buffer) == Lisp_Int)
X  #ifdef VMS
X***************
X*** 162,168 ****
X--- 176,186 ----
X  #endif /* not VMS */
X    else
X      {
X+ #ifdef atarist
X+       fd[0] = fd[1] = -2;
X+ #else
X        pipe (fd);
X+ #endif
X  #if 0
X        /* Replaced by close_process_descs */
X        set_exclusive_use (fd[0]);
X***************
X*** 183,218 ****
X--- 201,248 ----
X      env = environ;
X  #endif /* MAINTAIN_ENVIRONMENT */
X
X+ #ifndef atarist
X      pid = vfork ();
X+ #endif
X  #ifdef BSD4_1
X      /* cause SIGCHLD interrupts to look for this pid. */
X      synch_process_pid = pid;
X  #endif /* BSD4_1 */
X
X+ #ifndef atarist
X      if (pid == 0)
X+ #endif
X        {
X+ #ifndef atarist
X    if (fd[0] >= 0)
X      close (fd[0]);
X+ #endif
X    child_setup (filefd, fd1, fd1, new_argv, env);
X        }
X
X      environ = save_environ;
X
X+ #ifndef atarist
X      close (filefd);
X      close (fd1);
X+ #endif
X    }
```

```
X
X+ #ifndef atarist
X    if (pid < 0)
X      {
X        close (fd[0]);
X        report_file_error ("Doing vfork", Qnil);
X      }
X+ #endif
X
X    if (XTYPE (buffer) == Lisp_Int)
X      {
X+ #ifndef atarist
X  #ifndef subprocesses
X        wait_without_blocking ();
X  #endif subprocesses
X+ #endif
X        return Qnil;
X      }
X
X***************
X*** 229,234 ****
X--- 259,268 ----
X    {
X      register int nread;
X
X+ #ifdef atarist
X+     CHECK_STRING (Vshell_tmpfile, 0);
X+     fd[0] = open(XSTRING (Vshell_tmpfile)->data, O_RDONLY);
X+ #endif
X      while ((nread = read (fd[0], buf, sizeof buf)) > 0)
X        {
X    immediate_quit = 0;
X***************
X*** 239,248 ****
X--- 273,288 ----
X    immediate_quit = 1;
X    QUIT;
X        }
X+ #ifdef atarist
X+     close(fd[0]);
X+     unlink(XSTRING (Vshell_tmpfile)->data);
X+ #endif
X    }
X
X+ #ifndef atarist
X    /* Wait for it to terminate, unless it already has.  */
X    wait_for_termination (pid);
X+ #endif
X
X    immediate_quit = 0;
X
X***************
X*** 299,308 ****
```

```
X--- 339,354 ----
X     ENV is the environment */
X
X  child_setup (in, out, err, new_argv, env)
X+ #ifdef atarist
X+       char *in;
X+       int out, err;
X+ #else
X        int in, out, err;
X+ #endif
X        register char **new_argv;
X        char **env;
X  {
X+ #ifndef atarist
X    register int pid = getpid();
X
X    setpriority (PRIO_PROCESS, pid, 0);
X***************
X*** 311,316 ****
X--- 357,363 ----
X    /* Close Emacs's descriptors that this process should not have.  */
X    close_process_descs ();
X  #endif
X+ #endif
X
X    /* Note that use of alloca is always safe here.  It's obvious for systems
X       that do not have true vfork or that have true (stack) alloca.
X***************
X*** 358,363 ****
X--- 405,411 ----
X    }
X  #endif /* Not MAINTAIN_ENVIRONMENT */
X
X+ #ifndef atarist
X    close (0);
X    close (1);
X    close (2);
X***************
X*** 375,380 ****
X--- 423,429 ----
X    setpgrp (pid, pid);
X  #endif /* USG */
X    setpgrp_of_tty (pid);
X+ #endif
X
X  #ifdef vipc
X    something missing here;
X***************
X*** 384,394 ****
X--- 433,472 ----
X       to pass this environment is to set environ.  Our caller
X       is responsible for restoring the ambient value of environ.  */
X    environ = env;
```

```
X+ #ifdef atarist
X+   {
X+     register int i, len;
X+     char *argument;
X+
X+     for(i = len = 1; new_argv[i]; i++) len += strlen(new_argv[i]) + 1;
X+     if(strcmp(in,"/dev/null")) len += 2 + strlen(in);
X+     if (!NULL (Vshell_tmpfile)) {
X+       CHECK_STRING (Vshell_tmpfile, 0);
X+       len += 3 + strlen(XSTRING (Vshell_tmpfile)->data);
X+     } else len += 17;
X+     argument = (char *) alloca(len);
X+     *argument = '\0';
X+     for(i = 0; new_argv[i]; i++) {
X+       strcat(argument,new_argv[i]);
X+       strcat(argument," ");
X+     }
X+     if(strcmp(in,"/dev/null")) {
X+       strcat(argument,"< ");
X+       strcat(argument,in);
X+     }
X+     if (!NULL (Vshell_tmpfile)) {
X+       strcat(argument," > ");
X+       strcat(argument,XSTRING (Vshell_tmpfile)->data);
X+     } else strcat(argument, " ==> SHELL_OUTPUT");
X+     system(argument);
X+   }
X+ #else
X    execvp (new_argv[0], new_argv);
X
X    write (1, "Couldn't exec the program ", 26);
X    write (1, new_argv[0], strlen (new_argv[0]));
X    _exit (1);
X+ #endif
X  }
X

X  init_callproc ()
X***************
X*** 397,402 ****
X--- 475,483 ----
X    extern char **environ;
X    register char **envp;
X
X+ #ifdef atarist
X+   Vshell_tmpfile = build_string ("GNUemacs.tmp");
X+ #endif
X    Vexec_path = decode_env_path ("==", PATH_EXEC);
X    Vexec_directory = Ffile_name_as_directory (Fcar (Vexec_path));
X    Vexec_path = nconc2 (decode_env_path ("PATH", ""), Vexec_path);
X***************
X*** 437,442 ****
X--- 518,529 ----
```

```
X    DEFVAR_LISP ("exec-directory", &Vexec_directory,
X      "Directory that holds programs that come with GNU Emacs,\n\
X  intended for Emacs to invoke.");
X+
X+ #ifdef atarist
X+   DEFVAR_LISP ("shell-tmpfile", &Vshell_tmpfile,
X+     "Filename used for pipe simulation\n\
X+ or nil (use variable redirection into SHELL_OUTPUT).");
X+ #endif
X
X  #ifndef MAINTAIN_ENVIRONMENT
X    DEFVAR_LISP ("process-environment", &Vprocess_environment,
Xdiff -c original/config.h ./config.h
X*** original/config.h  Mon Jul 17 20:52:31 1989
X--- ./config.h    Sat Jul  8 16:46:07 1989
X***************
X*** 24,30 ****
X     See the file ../etc/MACHINES for a list of systems and
X     the names of the s- files to use for them.
X     See s-template.h for documentation on writing s- files.  */
X! #include "s-bsd4-3.h"
X
X  /* Include here a m- file that describes the machine and system you use.
X     See the file ../etc/MACHINES for a list of machines and
X--- 24,30 ----
X     See the file ../etc/MACHINES for a list of systems and
X     the names of the s- files to use for them.
X     See s-template.h for documentation on writing s- files.  */
X! #include "s-tos.h"
X
X  /* Include here a m- file that describes the machine and system you use.
X     See the file ../etc/MACHINES for a list of machines and
X***************
X*** 31,37 ****
X     the names of the m- files to use for them.
X     See m-template.h for info on what m- files should define.
X     */
X! #include "m-vax.h"
X
X  /* Load in the conversion definitions if this system
X     needs them and the source file being compiled has not
X--- 31,37 ----
X     the names of the m- files to use for them.
X     See m-template.h for info on what m- files should define.
X     */
X! #include "m-atari.h"
X
X  /* Load in the conversion definitions if this system
X     needs them and the source file being compiled has not
X***************
X*** 61,69 ****
X--- 61,71 ----
X
```

```
X  #ifndef VMS
X  #ifndef USG
X+ #ifndef atarist
X  #define subprocesses
X  #endif
X  #endif
X+ #endif
X
X  /* Define USER_FULL_NAME to return a string
X   that is the user's full name.
X***************
X*** 80,86 ****
X  /* Define AMPERSAND_FULL_NAME if you use the convention
X    that & in the full name stands for the login id.  */
X
X! #define AMPERSAND_FULL_NAME
X
X  /* Maximum screen width we handle. */
X
X--- 82,88 ----
X  /* Define AMPERSAND_FULL_NAME if you use the convention
X    that & in the full name stands for the login id.  */
X
X! /* #define AMPERSAND_FULL_NAME */
X
X  /* Maximum screen width we handle. */
X
Xdiff -c original/dired.c ./dired.c
X*** original/dired.c   Mon Jul 17 20:52:34 1989
X--- ./dired.c     Sat Jul  8 16:46:08 1989
X***************
X*** 207,216 ****
X       If nothing found then try again with passcount = 1, don't ignore them.
X       If looking for all completions, start with passcount = 1,
X       so always take even the ignored ones.  */
X    for (passcount = !!all_flag; NULL (bestmatch) && passcount < 2; passcount++)
X      {
X!       if (!(d = opendir (XSTRING (Fdirectory_file_name (dirname))->data)))
X!   report_file_error ("Opening directory", Fcons (dirname, Qnil));
X
X        /* Loop reading blocks */
X        /* (att3b compiler bug requires do a null comparison this way) */
X--- 207,217 ----
X       If nothing found then try again with passcount = 1, don't ignore them.
X       If looking for all completions, start with passcount = 1,
X       so always take even the ignored ones.  */
X+   if (!(d = opendir (XSTRING (Fdirectory_file_name (dirname))->data)))
X+     report_file_error ("Opening directory", Fcons (dirname, Qnil));
X    for (passcount = !!all_flag; NULL (bestmatch) && passcount < 2; passcount++)
X      {
X!       rewinddir(d);
X
X        /* Loop reading blocks */
```

```
X        /* (att3b compiler bug requires do a null comparison this way) */
X***************
X*** 309,316 ****
X         }
X        }
X    }
X-        closedir (d);
X      }
X
X    if (all_flag || NULL (bestmatch))
X      return bestmatch;
X--- 310,317 ----
X         }
X        }
X    }
X      }
X+   closedir (d);
X
X    if (all_flag || NULL (bestmatch))
X      return bestmatch;
X***************
X*** 414,420 ****
X--- 415,425 ----
X    else                    /* if we can't tell, assume worst */
X      values[9] = Qt;
X  #else                     /* file gid will be egid */
X+ #ifdef atarist
X+   values[9] = Qt;
X+ #else
X    values[9] = (s.st_gid != getegid ()) ? Qt : Qnil;
X+ #endif
X  #endif/* BSD4_2 (or BSD4_3) */
X  #ifdef BSD4_3
X  #undef BSD4_2 /* ok, you can look again without throwing up */
Xdiff -c original/dispnew.c ./dispnew.c
X*** original/dispnew.c Mon Jul 17 20:52:35 1989
X--- ./dispnew.c   Mon Jul 17 20:47:47 1989
X***************
X*** 1361,1366 ****
X--- 1361,1369 ----
X    terminal_type = (char *) getenv ("TERM");
X    if (!terminal_type)
X      {
X+ #ifdef atarist
X+       terminal_type = "atari";
X+ #endif /* atarist */
X  #ifdef VMS
X        fprintf (stderr, "Please specify your terminal type.\n\
X  For types defined in VMS, use  set term /device=TYPE.\n\
X***************
X*** 1369,1375 ****
X--- 1372,1382 ----
X  #else
```

```
X        fprintf (stderr, "Please set the environment variable TERM; see tset(1).
\n");
X  #endif
X+ #ifdef atarist
X+       fprintf (stderr, "I assume that your terminal type is \"%s\".\n",
terminal_type);
X+ #else
X        exit (1);
X+ #endif /* atarist */
X      }
X    term_init (terminal_type);
X
Xdiff -c original/editfns.c ./editfns.c
X*** original/editfns.c Mon Jul 17 20:52:37 1989
X--- ./editfns.c   Mon Jul 17 20:47:46 1989
X***************
X*** 20,26 ****
X--- 20,28 ----
X
X
X  #include "config.h"
X+ #ifndef atarist
X  #include <pwd.h>
X+ #endif
X  #include "lisp.h"
X  #include "buffer.h"
X  #include "window.h"
X***************
X*** 40,46 ****
X--- 42,50 ----
X  init_editfns ()
X  {
X    register char *p;
X+ #ifndef atarist
X    struct passwd *pw; /* password entry for the current user */
X+ #endif
X    extern char *index ();
X
X    /* Set up system_name even when dumping.  */
X***************
X*** 66,76 ****
X--- 70,84 ----
X      return;
X  #endif /* not CANNOT_DUMP */
X
X+ #ifndef atarist
X    pw = (struct passwd *) getpwuid (getuid ());
X    if (pw == 0)
X+ #endif atarist
X      strcpy (user_real_name, "unknown");
X+ #ifndef atarist
X    else
X      strncpy (user_real_name, pw->pw_name, sizeof user_real_name);
```

```
X+ #endif
X
X    user_name = (char *) getenv ("USER");
X    if (!user_name)
X***************
X*** 78,96 ****
X--- 86,113 ----
X    if (!user_name)
X      user_name = user_real_name;
X
X+ #ifndef atarist
X    if (strcmp (user_name, user_real_name))
X      pw = (struct passwd *) getpwnam (user_name);
X+ #endif
X
X  #ifndef AMPERSAND_FULL_NAME
X+ #ifndef atarist
X    if (pw == 0)
X+ #endif
X      strcpy (user_full_name, "unknown");
X+ #ifndef atarist
X    else
X      strncpy (user_full_name, USER_FULL_NAME, sizeof user_full_name);
X    p = index (user_full_name, ',');
X    if (p) *p = 0;
X+ #endif
X  #else
X+ #ifndef atarist
X    if (pw == 0)
X+ #endif
X      p = "unknown";
X+ #ifndef atarist
X    else
X      p = USER_FULL_NAME;
X    {
X***************
X*** 122,127 ****
X--- 139,145 ----
X        }
X      *q = 0;
X    }
X+ #endif
X  #endif /* AMPERSAND_FULL_NAME */
X  }
X

X***************
X*** 414,419 ****
X--- 432,438 ----
X    return build_string (user_real_name);
X  }
X
X+ #ifndef atarist
```

```
X  DEFUN ("user-uid", Fuser_uid, Suser_uid, 0, 0, 0,
X    "Return the effective uid of Emacs, as an integer.")
X    ()
X***************
X*** 427,432 ****
X--- 446,452 ----
X  {
X    return make_number (getuid ());
X  }
X+ #endif
X
X  DEFUN ("user-full-name", Fuser_full_name, Suser_full_name, 0, 0, 0,
X    "Return the full name of the user logged in, as a string.")
X***************
X*** 451,456 ****
X--- 471,500 ----
X    tem [24] = 0;
X    return build_string (tem);
X  }
X+ #ifdef atarist
X+ DEFUN ("time-to-string", Ftime_to_string, Stime_to_string, 1, 1, 0,
X+   "Convert arg TIME (a list of two integers) to a string describing that
time.")
X+   (t)
X+      register Lisp_Object t;
X+ {
X+   long n;
X+   register Lisp_Object tmp;
X+   register char *tem;
X+
X+   CHECK_LIST (t, 0);
X+   tmp = Fcdr (t);
X+   CHECK_LIST (tmp, 1);
X+   tmp = Fcar (tmp);
X+   CHECK_NUMBER (tmp, 1);
X+   n = XINT (tmp);
X+   tmp = Fcar (t);
X+   CHECK_NUMBER (tmp, 0);
X+   n |= XINT (tmp) << 16;
X+   tem = (char *) ctime (&n);
X+   tem[24] = 0;
X+   return build_string (tem);
X+ }
X+ #endif /* atarist */
X

X  DEFUN ("insert", Finsert, Sinsert, 0, MANY, 0,
X    "Any number of args, strings or chars.  Insert them after point, moving point
forward.")
X***************
X*** 1013,1022 ****
X--- 1057,1071 ----
X
```

```
X    defsubr (&Suser_login_name);
X    defsubr (&Suser_real_login_name);
X+ #ifndef atarist
X    defsubr (&Suser_uid);
X    defsubr (&Suser_real_uid);
X+ #endif atarist
X    defsubr (&Suser_full_name);
X    defsubr (&Scurrent_time_string);
X+ #ifdef atarist
X+   defsubr (&Stime_to_string);
X+ #endif /* atarist */
X    defsubr (&Ssystem_name);
X    defsubr (&Smessage);
X    defsubr (&Sformat);
Xdiff -c original/emacs.c ./emacs.c
X*** original/emacs.c   Mon Jul 17 20:52:37 1989
X--- ./emacs.c     Sat Jul  8 16:46:10 1989
X***************
X*** 171,176 ****
X--- 171,177 ----
X    int skip_args = 0;
X    extern int errno;
X    extern void malloc_warning ();
X+   extern void puts ();
X
X  #ifdef VMS
X  #ifdef LINK_CRTL_SHARE
X***************
X*** 210,221 ****
X      default_acl (USE_DEFACL);
X  #endif /* APOLLO */
X
X  #ifdef APOLLO            /* Reserve memory space for sbrk to get */
X    set_sbrk_size (4000000);
X  #else /* not APOLLO */
X  #ifndef SYSTEM_MALLOC
X    /* Arrange for warnings when nearly out of space.  */
X!   malloc_init (0, malloc_warning);
X  #endif
X  #endif /* not APOLLO */
X
X--- 212,223 ----
X      default_acl (USE_DEFACL);
X  #endif /* APOLLO */
X
X  #ifdef APOLLO            /* Reserve memory space for sbrk to get */
X    set_sbrk_size (4000000);
X  #else /* not APOLLO */
X  #ifndef SYSTEM_MALLOC
X    /* Arrange for warnings when nearly out of space.  */
X!   malloc_init (0, puts); /* until everything is initialized */
X  #endif
X  #endif /* not APOLLO */
```

```
X
X***************
X*** 425,430 ****
X--- 440,446 ----
X        keys_of_window ();
X      }
X
X+   malloc_init (0, malloc_warning);
X    if (!initialized)
X      {
X        /* Handle -l loadup-and-dump, args passed by Makefile. */
X***************
X*** 558,564 ****
X
X  #endif /* not CANNOT_DUMP */
X

X! #ifdef VMS
X  #define SEPCHAR ','
X  #else
X  #define SEPCHAR ':'
X--- 579,585 ----
X
X  #endif /* not CANNOT_DUMP */
X

X! #if defined(VMS) || defined(atarist)
X  #define SEPCHAR ','
X  #else
X  #define SEPCHAR ':'
Xdiff -c original/environ.c ./environ.c
X*** original/environ.c Mon Jul 17 20:52:38 1989
X--- ./environ.c   Mon Jul 17 20:47:45 1989
X***************
X*** 255,260 ****
X--- 255,295 ----
X
X  #endif /* dead code */
X
X+ #ifdef atarist
X+ DEFUN ("getvar", Fgetvar, Sgetvar, 1, 2, "sEnvironment variable: \np",
X+   "Get the contents of the shell variable VAR and set as an environment\n
X+ variable with the same name. Return the value of VAR as a string.\n
X+ VAR is a string, the name of the variable.")
X+   (str, interactivep)
X+      Lisp_Object str, interactivep;
X+ {
X+   Lisp_Object val;
X+   unsigned char *value;
X+   extern unsigned char *m_getenv();
X+
X+   CHECK_STRING (str, 0);
X+   value = m_getenv(XSTRING (str)->data);
```

```
X+   if (value)
X+     val = build_string(value);
X+   else
X+     val = Qnil;
X+
X+   if (!NULL (val)) {
X+     CHECK_STRING (val, 0);
X+     set_environment_alist (str, val);
X+   }
X+
X+   if (!NULL (interactivep))
X+     {
X+       if (NULL (val))
X+   message ("%s not defined in shell", XSTRING (str)->data);
X+       else
X+   message ("\"%s\"", XSTRING (val)->data);
X+     }
X+   return val;
X+ }
X+ #endif /* atarist */
X+
X

X  DEFUN ("getenv", Fgetenv, Sgetenv, 1, 2, "sEnvironment variable: \np",
X    "Return the value of environment variable VAR, as a string.\n\
X***************
X*** 267,273 ****
X  {
X    Lisp_Object val;
X
X!   if (str == Qt)         /* If arg is t, return whole environment */
X      return (Fcopy_alist (Venvironment_alist));
X
X    CHECK_STRING (str, 0);
X--- 302,308 ----
X  {
X    Lisp_Object val;
X
X!   if (EQ(str, Qt))       /* If arg is t, return whole environment */
X      return (Fcopy_alist (Venvironment_alist));
X
X    CHECK_STRING (str, 0);
X***************
X*** 305,310 ****
X--- 340,346 ----
X  syms_of_environ ()
X  {
X    staticpro (&Venvironment_alist);
X+   defsubr (&Sgetvar);
X    defsubr (&Ssetenv);
X    defsubr (&Sgetenv);
X  }
Xdiff -c original/fileio.c ./fileio.c
```

```
X*** original/fileio.c  Mon Jul 17 20:52:40 1989
X--- ./fileio.c    Mon Jul 17 20:47:45 1989
X***************
X*** 21,27 ****
X--- 21,29 ----
X
X  #include <sys/types.h>
X  #include <sys/stat.h>
X+ #ifndef atarist
X  #include <pwd.h>
X+ #endif
X  #include <ctype.h>
X  #include <sys/dir.h>
X  #include <errno.h>
X***************
X*** 65,70 ****
X--- 67,76 ----
X  #include <errnet.h>
X  #endif
X
X+ #ifdef atarist
X+ #include <file.h>
X+ #endif
X+
X  #define min(a, b) ((a) < (b) ? (a) : (b))
X  #define max(a, b) ((a) > (b) ? (a) : (b))
X
X***************
X*** 430,436 ****
X--- 436,444 ----
X    register unsigned char *newdir, *p, *o;
X    int tlen;
X    unsigned char *target;
X+ #ifndef atarist
X    struct passwd *pw;
X+ #endif
X    int lose;
X  #ifdef VMS
X    unsigned char * colon = 0;
X***************
X*** 576,581 ****
X--- 584,592 ----
X        {
X    if (!(newdir = (unsigned char *) egetenv ("HOME")))
X      newdir = (unsigned char *) "";
X+ #ifdef atarist
X+   else tos_to_unix(newdir);
X+ #endif /* atarist */
X    nm++;
X  #ifdef VMS
X    nm++;              /* Don't leave the slash in nm.  */
X***************
X*** 592,600 ****
```

```
X--- 603,618 ----
X    bcopy ((char *) nm, o, p - nm);
X    o [p - nm] = 0;
X
X+ #ifdef atarist
X+   newdir = "/dev/A";
X+   newdir[5] = toupper(o[p - nm - 1]);
X+         if ((newdir[5] < 'A') || (newdir[5] > 'P'))
X+     error ("\"%s\" isn't an allowed drive", o + 1);
X+ #else
X    pw = (struct passwd *) getpwnam (o + 1);
X    if (!pw)
X      error ("\"%s\" isn't a registered user", o + 1);
X+ #endif
X
X  #ifdef VMS
X    nm = p + 1;        /* skip the terminator */
X***************
X*** 601,607 ****
X--- 619,627 ----
X  #else
X    nm = p;
X  #endif /* VMS */
X+ #ifndef atarist
X    newdir = (unsigned char *) pw -> pw_dir;
X+ #endif
X        }
X
X    if (nm[0] != '/'
X***************
X*** 756,762 ****
X
X    for (p = nm; p != endp; p++)
X      {
X!       if ((p[0] == '~' ||
X  #ifdef APOLLO
X       /* // at start of file name is meaningful in Apollo system */
X       (p[0] == '/' && p - 1 != nm)
X--- 776,782 ----
X
X    for (p = nm; p != endp; p++)
X      {
X!       if (((p[0] == '~' ||
X  #ifdef APOLLO
X       /* // at start of file name is meaningful in Apollo system */
X       (p[0] == '/' && p - 1 != nm)
X***************
X*** 772,777 ****
X--- 792,798 ----
X  #ifdef VMS
X      )
X  #endif /* VMS */
X+   )
```

```
X    {
X      nm = p;
X      substituted = 1;
X***************
X*** 801,811 ****
X        substituted = 1;
X        continue;
X      }
X!   else if (*p == '{')
X      {
X        o = ++p;
X!       while (p != endp && *p != '}') p++;
X!       if (*p != '}') goto missingclose;
X        s = p;
X      }
X    else
X--- 822,832 ----
X        substituted = 1;
X        continue;
X      }
X!   else if (*p == '{') /*}*/
X      {
X        o = ++p;
X!       /*{*/while (p != endp && *p != '}') p++;
X!       /*{*/if (*p != '}') goto missingclose;
X        s = p;
X      }
X    else
X***************
X*** 860,866 ****
X      {
X        o = ++p;
X        while (p != endp && *p != '}') p++;
X!       if (*p != '}') goto missingclose;
X        s = p++;
X      }
X    else
X--- 881,887 ----
X      {
X        o = ++p;
X        while (p != endp && *p != '}') p++;
X!       /*{*/ if (*p != '}') goto missingclose;
X        s = p++;
X      }
X    else
X***************
X*** 896,902 ****
X    /* If /~ or // appears, discard everything through first slash. */
X
X    for (p = xnm; p != x; p++)
X!     if ((p[0] == '~' ||
X  #ifdef APOLLO
X     /* // at start of file name is meaningful in Apollo system */
```

```
X     (p[0] == '/' && p - 1 != xnm)
X--- 917,923 ----
X    /* If /~ or // appears, discard everything through first slash. */
X
X    for (p = xnm; p != x; p++)
X!     if (((p[0] == '~' ||
X  #ifdef APOLLO
X     /* // at start of file name is meaningful in Apollo system */
X     (p[0] == '/' && p - 1 != xnm)
X***************
X*** 905,910 ****
X--- 926,932 ----
X  #endif /* not APOLLO */
X     )
X    && p != nm && p[-1] == '/')
X+   )
X        xnm = p;
X
X    return make_string (xnm, x - xnm);
X***************
X*** 912,918 ****
X   badsubst:
X    error ("Bad format environment-variable substitution");
X   missingclose:
X!   error ("Missing \"}\" in environment-variable substitution");
X   badvar:
X    error ("Substituting nonexistent environment variable \"%s\"", target);
X
X--- 934,940 ----
X   badsubst:
X    error ("Bad format environment-variable substitution");
X   missingclose:
X!   /*{*/ error ("Missing \"}\" in environment-variable substitution");
X   badvar:
X    error ("Substituting nonexistent environment variable \"%s\"", target);
X
X***************
X*** 1100,1105 ****
X--- 1122,1128 ----
X    return Qnil;
X  }
X
X+ #ifndef atarist
X  DEFUN ("add-name-to-file", Fadd_name_to_file, Sadd_name_to_file, 2, 3,
X    "fAdd name to file: \nFName to add to %s: \np",
X    "Give FILE additional name NEWNAME.  Both args strings.\n\
X***************
X*** 1136,1141 ****
X--- 1159,1165 ----
X
X    return Qnil;
X  }
X+ #endif
```

```
X
X  #ifdef S_IFLNK
X  DEFUN ("make-symbolic-link", Fmake_symbolic_link, Smake_symbolic_link, 2, 3,
X***************
X*** 1682,1687 ****
X--- 1706,1712 ----
X    }
X      }
X
X+ #ifndef atarist
X  #ifndef USG
X  #ifndef VMS
X  #ifndef BSD4_1
X***************
X*** 1692,1697 ****
X--- 1717,1723 ----
X  #endif
X  #endif
X  #endif
X+ #endif
X
X  #if 0
X    /* Spurious "file has changed on disk" warnings have been
X***************
X*** 2028,2033 ****
X--- 2058,2066 ----
X
X    /* If dir starts with user's homedir, change that to ~. */
X    homedir = (char *) egetenv ("HOME");
X+ #ifdef atarist
X+   tos_to_unix(homedir);
X+ #endif
X    if (homedir != 0
X        && XTYPE (dir) == Lisp_String
X        && !strncmp (homedir, XSTRING (dir)->data, strlen (homedir))
X***************
X*** 2103,2109 ****
X--- 2136,2144 ----
X    defsubr (&Scopy_file);
X    defsubr (&Sdelete_file);
X    defsubr (&Srename_file);
X+ #ifndef atarist
X    defsubr (&Sadd_name_to_file);
X+ #endif
X  #ifdef S_IFLNK
X    defsubr (&Smake_symbolic_link);
X  #endif /* S_IFLNK */
Xdiff -c original/fns.c ./fns.c
X*** original/fns.c    Mon Jul 17 20:52:42 1989
X--- ./fns.c  Sat Jul  8 16:46:13 1989
X***************
X*** 47,53 ****
X--- 47,57 ----
```

```
X  #ifndef NLIST_STRUCT
X  #include <a.out.h>
X  #else /* NLIST_STRUCT */
X+ #ifdef atarist
X+ #include <gnu-out.h>
X+ #else
X  #include <nlist.h>
X+ #endif
X  #endif /* NLIST_STRUCT */
X  #endif /* not VMS */
X  #endif /* LOAD_AVE_TYPE */
Xdiff -c original/keyboard.c ./keyboard.c
X*** original/keyboard.c    Mon Jul 17 20:52:45 1989
X--- ./keyboard.c  Sat Jul  8 16:46:47 1989
X***************
X*** 42,47 ****
X--- 42,50 ----
X  #include <termio.h>
X  #include <fcntl.h>
X  #else /* not USG */
X+ #ifdef atarist
X+ #include <fcntl.h>
X+ #endif
X  #ifndef VMS
X  #include <sys/ioctl.h>
X  #endif /* not VMS */
X***************
X*** 176,186 ****
X--- 179,197 ----
X  int flow_control;
X
X  #ifndef BSD4_1
X+ #ifdef atarist
X+ #define sigfree()
X+ #define sigholdx(sig)
X+ #define sigblockx(sig)
X+ #define sigunblockx(sig)
X+ #define sigpausex(sig)
X+ #else
X  #define sigfree() sigsetmask (0)
X  #define sigholdx(sig) sigsetmask (1 << ((sig) - 1))
X  #define sigblockx(sig) sigblock (1 << ((sig) - 1))
X  #define sigunblockx(sig) sigblock (0)
X  #define sigpausex(sig) sigpause (0)
X+ #endif
X  #endif /* not BSD4_1 */
X
X  #ifdef BSD4_1
X***************
X*** 1047,1053 ****
X      }
X
X  #else /* no FIONREAD */
```

```
X! #ifdef USG
X    if (kbd_count)
X      abort ();
X
X--- 1058,1064 ----
X      }
X
X  #else /* no FIONREAD */
X! #if defined(USG) || defined(atarist)
X    if (kbd_count)
X      abort ();
X
X***************
X*** 1101,1107 ****
X    extern int select_alarmed;
X  #endif
X
X! #ifdef USG
X    /* USG systems forget handlers when they are used;
X       must reestablish each time */
X    signal (signo, input_available_signal);
X--- 1112,1118 ----
X    extern int select_alarmed;
X  #endif
X
X! #if defined(USG) || defined(atarist)
X    /* USG systems forget handlers when they are used;
X       must reestablish each time */
X    signal (signo, input_available_signal);
X***************
X*** 1682,1688 ****
X    char c;
X    extern Lisp_Object Vwindow_system;
X
X! #ifdef USG
X    /* USG systems forget handlers when they are used;
X       must reestablish each time */
X    signal (SIGINT, interrupt_signal);
X--- 1693,1699 ----
X    char c;
X    extern Lisp_Object Vwindow_system;
X
X! #if defined(USG) || defined(atarist)
X    /* USG systems forget handlers when they are used;
X       must reestablish each time */
X    signal (SIGINT, interrupt_signal);
X***************
X*** 1805,1811 ****
X    if (!noninteractive)
X      {
X        signal (SIGINT, interrupt_signal);
X! #ifdef USG
X        /* On USG systems, C-g is set up for both SIGINT and SIGQUIT
```

```
X     and we can't tell which one it will give us.  */
X        signal (SIGQUIT, interrupt_signal);
X--- 1816,1822 ----
X    if (!noninteractive)
X      {
X        signal (SIGINT, interrupt_signal);
X! #if defined(USG) || defined(atarist)
X        /* On USG systems, C-g is set up for both SIGINT and SIGQUIT
X     and we can't tell which one it will give us.  */
X        signal (SIGQUIT, interrupt_signal);
Xdiff -c original/keymap.c ./keymap.c
X*** original/keymap.c  Mon Jul 17 20:52:46 1989
X--- ./keymap.c    Sat Jul  8 16:46:48 1989
X***************
X*** 778,785 ****
X
```

```
From -8994037487077101365
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,7f59ce649233d586,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-08-08 15:13:17 PST
Path: utzoo!utgpu!watmath!att!pacbell!ames!sun-barr!newstop!sun!grapevine!
koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i069: emacs.diff -- Diffs to GNU-EMACS 18.52 for Atari  part02/02
Keywords: shar, diff
Message-ID: <34059@grapevine.uucp>
Date: 8 Aug 89 22:13:17 GMT
Sender: news@grapevine.uucp
Lines: 935
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: uunet.UU.NET!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 2, Issue 69
Archive-name: emacs.diff/part02

[Append this to part 1, and edit the result so that there's an uninterrupted
 column of Xs along the left of the file. -sg]

X    for (; !NULL (maps); maps = Fcdr (maps))
X      {
X!       register this = Fcar (Fcar (maps)); /* Key sequence to reach map */
X!       register map = Fcdr (Fcar (maps)); /* The map that it reaches */
X        register int i = 0;
X
X        if (CONSP (map))
X--- 778,787 ----
X
X    for (; !NULL (maps); maps = Fcdr (maps))
X      {
```

```
X!       register Lisp_Object this = Fcar (Fcar (maps));
X!            /* Key sequence to reach map */
X!       register Lisp_Object map = Fcdr (Fcar (maps));
X!            /* The map that it reaches */
X        register int i = 0;
X
X        if (CONSP (map))
X***************
X*** 1099,1105 ****
X    {
X      Lisp_Object tem;
X      if (NULL (kludge)) kludge = build_string ("x");
X!     XSTRING (kludge)->data[0] = XINT (i);
X      tem = Flookup_key (shadow, kludge);
X      if (!NULL (tem)) continue;
X    }
X--- 1101,1107 ----
X    {
X      Lisp_Object tem;
X      if (NULL (kludge)) kludge = build_string ("x");
X!     XSTRING (kludge)->data[0] = i;
X      tem = Flookup_key (shadow, kludge);
X      if (!NULL (tem)) continue;
X    }
X***************
X*** 1167,1172 ****
X--- 1169,1176 ----
X    if (!NULL (tem))
X      apropos_accumulate = Fcons (symbol, apropos_accumulate);
X  }
X+
X+ static insert_first_line ();
X
X  static Lisp_Object
X  apropos1 (list)
Xdiff -c original/lisp.h ./lisp.h
X*** original/lisp.h   Mon Jul 17 20:52:47 1989
X--- ./lisp.h Sat Jul  8 16:46:49 1989
X***************
X*** 875,880 ****
X--- 875,881 ----
X
X  extern Lisp_Object Fmarker_position (), Fmarker_buffer ();
X  extern Lisp_Object Fcopy_marker ();
X+ extern Lisp_Object Fset_marker ();
X
X  /* Defined in fileio.c */
X
X***************
X*** 883,888 ****
X--- 884,891 ----
X  extern Lisp_Object Fexpand_file_name (), Ffile_name_nondirectory ();
X  extern Lisp_Object Fsubstitute_in_file_name ();
```

```
X  extern Lisp_Object Ffile_symlink_p ();
X+ extern Lisp_Object Fdirectory_file_name ();
X+ extern Lisp_Object Fverify_visited_file_modtime (), Ffile_exists_p ();
X
X  /* Defined in abbrev.c */
X
X***************
X*** 926,932 ****
X  extern Lisp_Object Fkey_description (), Fsingle_key_description ();
X  extern Lisp_Object Fwhere_is_internal ();
X  extern Lisp_Object access_keymap (), store_in_keymap ();
X! extern Lisp_Object get_keyelt (), get_keymap();
X
X  /* defined in indent.c */
X  extern Lisp_Object Fvertical_motion (), Findent_to (), Fcurrent_column ();
X--- 929,935 ----
X  extern Lisp_Object Fkey_description (), Fsingle_key_description ();
X  extern Lisp_Object Fwhere_is_internal ();
X  extern Lisp_Object access_keymap (), store_in_keymap ();
X! extern Lisp_Object get_keyelt (), get_keymap(), get_keymap_1();
X
X  /* defined in indent.c */
X  extern Lisp_Object Fvertical_motion (), Findent_to (), Fcurrent_column ();
X***************
X*** 969,974 ****
X--- 972,980 ----
X
X  /* defined in macros.c */
X  extern Lisp_Object Fexecute_kbd_macro ();
X+
X+ /* defined somewhere (but i don't have the nerve to find it */
X+ extern Lisp_Object Fcopy_alist ();
X
X  /* Nonzero means Emacs has already been initialized.
X     Used during startup to detect startup of dumped Emacs.  */
Xdiff -c original/lread.c ./lread.c
X*** original/lread.c   Mon Jul 17 20:52:48 1989
X--- ./lread.c     Sat Jul  8 16:46:49 1989
X***************
X*** 562,567 ****
X--- 565,572 ----
X

X  static int read_buffer_size;
X  static char *read_buffer;
X+
X+ static int read_escape ();
X
X  static Lisp_Object
X  read1 (readcharfun)
Xdiff -c original/paths.h ./paths.h
X*** original/paths.h   Mon Jul 17 20:53:06 1989
X--- ./paths.h     Sat Jul  8 16:46:52 1989
```

```
X***************
X*** 1,19 ****
X  /* The default search path for Lisp function "load".
X     This sets load-path.  */
X! #define PATH_LOADSEARCH "/usr/public/emacs/lisp"
X
X  /* the extra search path for programs to invoke.
X   This is appended to whatever the PATH environment variable says
X   to set the Lisp variable exec-path and the first file namein it
X    sets the Lisp variable exec-directory.  */
X! #define PATH_EXEC "/usr/public/emacs/etc"
X
X  /* the name of the directory that contains lock files
X   with which we record what files are being modified in Emacs.
X   This directory should be writable by everyone.
X   THE STRING MUST END WITH A SLASH!!!  */
X! #define PATH_LOCK "/usr/tmp/"
X
X  /* the name of the file !!!SuperLock!!! in the directory
X   specified by PATH_LOCK.  Yes, this is redundant.  */
X! #define PATH_SUPERLOCK "/usr/tmp/!!!SuperLock!!!"
X--- 1,19 ----
X  /* The default search path for Lisp function "load".
X     This sets load-path.  */
X! #define PATH_LOADSEARCH "/usr/lib/GNUemacs/lisp"
X
X  /* the extra search path for programs to invoke.
X   This is appended to whatever the PATH environment variable says
X   to set the Lisp variable exec-path and the first file namein it
X    sets the Lisp variable exec-directory.  */
X! #define PATH_EXEC "/usr/lib/GNUemacs/etc"
X
X  /* the name of the directory that contains lock files
X   with which we record what files are being modified in Emacs.
X   This directory should be writable by everyone.
X   THE STRING MUST END WITH A SLASH!!!  */
X! #define PATH_LOCK "/tmp/"
X
X  /* the name of the file !!!SuperLock!!! in the directory
X   specified by PATH_LOCK.  Yes, this is redundant.  */
X! #define PATH_SUPERLOCK "/tmp/GNUemacs.LCK"
Xdiff -c original/regex.c ./regex.c
X*** original/regex.c   Mon Jul 17 20:53:11 1989
X--- ./regex.c     Sat Jul  8 16:46:53 1989
X***************
X*** 967,972 ****
X--- 967,974 ----
X
X  int re_max_failures = 2000;
X
X+ static int bcmp_translate ();
X+
X  /* Match the pattern described by PBUFP
```

```
X     against data which is the virtual concatenation of STRING1 and STRING2.
X     SIZE1 and SIZE2 are the sizes of the two data strings.
Xdiff -c original/search.c ./search.c
X*** original/search.c  Mon Jul 17 20:53:21 1989
X--- ./search.c    Sat Jul  8 16:46:55 1989
X***************
X*** 945,950 ****
X--- 945,952 ----
X    return search_command (string, bound, noerror, count, 1, 1);
X  }
X

X+ static place ();
X+
X  DEFUN ("replace-match", Freplace_match, Sreplace_match, 1, 3, 0,
X    "Replace text matched by last search with NEWTEXT.\n\
X  If second arg FIXEDCASE is non-nil, do not alter case of replacement text.\n\
Xdiff -c original/syntax.c ./syntax.c
X*** original/syntax.c  Mon Jul 17 20:53:23 1989
X--- ./syntax.c    Sat Jul  8 16:46:55 1989
X***************
X*** 146,153 ****
X    (ch)
X       Lisp_Object ch;
X  {
X    CHECK_NUMBER (ch, 0);
X!   return make_number (syntax_code_spec[(int) SYNTAX (0xFF & XINT (ch))]);
X  }
X
X  /* This comment supplies the doc string for modify-syntax-entry,
X--- 149,159 ----
X    (ch)
X       Lisp_Object ch;
X  {
X+   int dummy_for_stupid_gcc;
X+
X    CHECK_NUMBER (ch, 0);
X!   dummy_for_stupid_gcc = (int) SYNTAX (0xFF & XINT (ch));
X!   return make_number (syntax_code_spec[dummy_for_stupid_gcc]);
X  }
X
X  /* This comment supplies the doc string for modify-syntax-entry,
X***************
X*** 1029,1039 ****
X--- 1043,1054 ----
X
X  */
X
X  DEFUN ("parse-partial-sexp", Fparse_partial_sexp, Sparse_partial_sexp, 2, 5, 0,
X    0 /* See immediately above */)
X    (from, to, targetdepth, stopbefore, oldstate)
X       Lisp_Object from, to, targetdepth, stopbefore, oldstate;
X  {
```

```
X+   Lisp_Object   dummy_for_stupid_gcc;
X    struct lisp_parse_state state;
X    int target;
X
X***************
X*** 1050,1063 ****
X                    target, !NULL (stopbefore), oldstate);
X
X    SetPoint (state.location);
X!
X    return Fcons (make_number (state.depth),
X       Fcons (state.prevlevelstart < 0 ? Qnil : make_number
(state.prevlevelstart),
X         Fcons (state.thislevelstart < 0 ? Qnil : make_number
(state.thislevelstart),
X!          Fcons (state.instring >= 0 ? make_number (state.instring) : Qnil,
X!         Fcons (state.incomment ? Qt : Qnil,
X!           Fcons (state.quoted ? Qt : Qnil,
X!               Fcons (make_number (state.mindepth), Qnil)))))));
X  }
X

X  init_syntax_once ()
X--- 1067,1080 ----
X                    target, !NULL (stopbefore), oldstate);
X
X    SetPoint (state.location);
X!   dummy_for_stupid_gcc = Fcons (state.instring >= 0 ? make_number
(state.instring) : Qnil,
X!         Fcons (state.incomment ? Qt : Qnil,
X!           Fcons (state.quoted ? Qt : Qnil,
X!               Fcons (make_number (state.mindepth), Qnil))));
X    return Fcons (make_number (state.depth),
X       Fcons (state.prevlevelstart < 0 ? Qnil : make_number
(state.prevlevelstart),
X         Fcons (state.thislevelstart < 0 ? Qnil : make_number
(state.thislevelstart),
X!          dummy_for_stupid_gcc)));
X  }
X

X  init_syntax_once ()
Xdiff -c original/sysdep.c ./sysdep.c
X*** original/sysdep.c  Mon Jul 17 20:53:25 1989
X--- ./sysdep.c    Mon Jul 17 20:47:48 1989
X***************
X*** 73,78 ****
X--- 73,81 ----
X  #ifdef USG
X  #include <fcntl.h>
X  #endif
X+ #ifdef atarist
X+ #include <fcntl.h>
```

```
X+ #endif
X  #endif /* not 4.1 bsd */
X
X  #ifdef BSD
X***************
X*** 119,124 ****
X--- 122,135 ----
X  #define TCSETAW TIOCSETN
X  #endif /* not VMS */
X  #endif /* not HAVE_TERMIO */
X+ #ifdef atarist
X+ #define kByte   * 1024
X+ extern long     _stksize = 64 kByte;
X+ #include <osbind.h>
X+ #define gethostname(a,b) strncpy(a,"Atari ST",b)
X+ #undef ANYP
X+ #define ANYP    META
X+ #endif
X
X  #ifdef USG
X  #include <sys/utsname.h>
X***************
X*** 351,359 ****
X--- 362,374 ----
X  #endif /* not BSD */
X  #else /* not subprocesses */
X  #ifndef BSD4_1
X+ #ifdef atarist
X+       break;
X+ #else
X        if (0 > kill (pid, 0))
X    break;
X        wait (0);
X+ #endif
X  #else /* BSD4_1 */
X        int status;
X        status = wait (0);
X***************
X*** 453,458 ****
X--- 468,482 ----
X
X  sys_suspend ()
X  {
X+ #ifdef atarist
X+   extern int    raise();
X+   struct sgttyb oldtty;
X+
X+   ioctl(0, TIOCGETP, &oldtty);
X+   raise(SIGTSTP);
X+   /* on Atari ST this is only needed for enabling META-Mode */
X+   ioctl(0, TIOCSETN, &oldtty);
X+ #else
X  #ifdef VMS
```

```
X    unsigned long parent_id;
X
X***************
X*** 536,543 ****
X--- 560,573 ----
X  #endif /* no USG_JOBCTRL */
X  #endif /* no SIGTSTP */
X  #endif /* not VMS */
X+ #endif /* not atarist */
X  }
X

X+ #ifdef atarist
X+
X+ int old_fcntl_flags;
X+
X+ #else
X  #ifdef F_SETFL
X
X  int old_fcntl_flags;
X***************
X*** 603,609 ****
X--- 633,658 ----
X  #endif /* STRIDE */
X  #endif /* FASYNC */
X  #endif /* F_SETFL */
X+ #endif /* not atarist */
X

X+ #ifdef atarist
X+ croak (badfunc)
X+      char *badfunc;
X+ {
X+   printf ("%s not yet implemented\r\n", badfunc);
X+ }
X+
X+ request_sigio ()
X+ {
X+   croak ("request_sigio");
X+ }
X+
X+ unrequest_sigio ()
X+ {
X+   croak ("unrequest_sigio");
X+ }
X+ #endif
X+
X  TERMINAL old_gtty;       /* The initial tty mode bits */
X
X  int term_initted;        /* 1 if outer tty status has been recorded */
X***************
X*** 765,771 ****
X--- 814,822 ----
```

```
X  #endif /* TIOCGLTC */
X  #ifdef TIOCGETC
X       ioctl (0, TIOCGETC, &old_tchars);
X+ #ifndef atarist
X       ioctl (0, TIOCLGET, &old_lmode);
X+ #endif
X
X       /* Note: if not using CBREAK mode, it makes no difference how we set this
*/
X       tchars = new_tchars;
X***************
X*** 784,793 ****
X--- 835,848 ----
X  #define LNOFLSH 0100000
X  #endif
X
X+ #ifndef atarist
X       lmode = LDECCTQ | LLITOUT | LPASS8 | LNOFLSH | old_lmode;
X+ #endif
X
X       ioctl (0, TIOCSETC, &tchars);
X+ #ifndef atarist
X       ioctl (0, TIOCLSET, &lmode);
X+ #endif
X  #endif /* TIOCGETC */
X  #ifdef TIOCGLTC
X       ioctl (0, TIOCSLTC, &new_ltchars);
X***************
X*** 845,850 ****
X--- 900,922 ----
X     Store number of lines into *heightp and width into *widthp.
X     If zero or a negative number is stored, the value is not valid.  */
X
X+ #ifdef atarist
X+ short *lineA_start;
X+
X+ static long
X+ linea0()
X+ {
X+   asm("movl a0, sp@-");
X+   asm("movl a1, sp@-");
X+   asm("movl a2, sp@-");
X+   asm(".word    0xA000");
X+   asm("movl a0, _lineA_start");
X+   asm("movl sp@+, a2");
X+   asm("movl sp@+, a1");
X+   asm("movl sp@+, a0");
X+ }
X+ #endif
X+
X  get_screen_size (widthp, heightp)
X       int *widthp, *heightp;
X  {
```

```
X***************
X*** 879,886 ****
X--- 951,964 ----
X    *widthp = sg.scr_wid;
X    *heightp = sg.scr_len;
X  #else /* system doesn't know size */
X+ #ifdef atarist
X+   linea0();
X+   *widthp  = lineA_start[-22] + 1;
X+   *heightp = lineA_start[-21] + 1;
X+ #else
X    *widthp = 0;
X    *heightp = 0;
X+ #endif
X  #endif /* system does not know size */
X  #endif /* not TIOCGWINSZ */
X  }
X***************
X*** 915,921 ****
X--- 993,1001 ----
X  #endif /* TIOCGLTC */
X  #ifdef TIOCGETC
X    ioctl (0, TIOCSETC, &old_tchars);
X+ #ifndef atarist
X    ioctl (0, TIOCLSET, &old_lmode);
X+ #endif
X  #endif /* TIOCGETC */
X  #ifdef F_SETFL
X  #ifdef F_SETOWN     /* F_SETFL does not imply existance of F_SETOWN */
X***************
X*** 935,941 ****
X--- 1015,1023 ----
X    SYS$QIOW (0, input_chan, IO$_SETMODE, &input_iosb, 0, 0,
X        &old_gtty.class, 12, 0, 0, 0, 0);
X  #else /* not VMS */
X+ #ifndef atarist
X    while (ioctl (0, TCSETAW, &old_gtty) < 0 && errno == EINTR);
X+ #endif
X  #endif /* not VMS */
X  }
X

X***************
X*** 1342,1348 ****
X--- 1424,1432 ----
X  {
X    int ravail = 0, orfds = 0, old_alarm;
X    extern int kbd_count;
X+ #ifndef atarist
X    extern int proc_buffered_char[];
X+ #endif
X  #ifndef subprocesses
X    int child_changed = 0;
```

```
X  #else
X***************
X*** 1394,1399 ****
X--- 1478,1484 ----
X  #ifdef FIONREAD
X              status = ioctl (fd, FIONREAD, &avail);
X  #else /* no FIONREAD */
X+ #ifndef atarist
X              /* Hoping it will return -1 if nothing available
X              or 0 if all 0 chars requested are read.  */
X              if (proc_buffered_char[fd] >= 0)
X***************
X*** 1404,1409 ****
X--- 1489,1495 ----
X               if (avail > 0)
X                 proc_buffered_char[fd] = buf;
X             }
X+ #endif
X  #endif /* no FIONREAD */
X            }
X          if (status >= 0 && avail > 0)
X***************
X*** 1428,1436 ****
X--- 1514,1531 ----
X         That makes instant wakeup for terminal input at least.  */
X      if (orfds & 1)
X        {
X+ #ifdef atarist
X+         /* since read cannot be interrupted (it's a trap in supervisor
X+         mode) on the ST, we get a possible timeout this way without
X+         loosing any input chars */
X+         old_fcntl_flags = fcntl (fileno(stdin), F_SETFL, O_NDELAY);
X+ #endif /* atarist*/
X          read_input_waiting ();
X          if (kbd_count)
X         select_alarmed = 1;
X+ #ifdef atarist
X+         fcntl (fileno(stdin), F_SETFL, old_fcntl_flags);
X+ #endif /* atarist*/
X        }
X      else
X        pause();
Xdiff -c original/termcap.c ./termcap.c
X*** original/termcap.c Mon Jul 17 20:53:26 1989
X--- ./termcap.c   Mon Jul 17 20:47:47 1989
X***************
X*** 436,442 ****
X--- 436,447 ----
X       it is the entry itself, but only if
X       the name the caller requested matches the TERM variable.  */
X
X+ #ifdef atarist
X+   /* the '?:' construct requires GNU C */
```

```
X+   if (tem && !filep && !strcmp (name, (char *) getenv ("TERM") ? : "atari"))
X+ #else
X    if (tem && !filep && !strcmp (name, getenv ("TERM")))
X+ #endif /* atarist */
X      {
X        indirect = tgetst1 (find_capability (tem, "tc"), 0);
X        if (!indirect)
Xdiff -c original/window.c ./window.c
X*** original/window.c  Mon Jul 17 20:53:34 1989
X--- ./window.c    Sat Jul  8 16:47:00 1989
X***************
X*** 217,228 ****
X    (window)
X       Lisp_Object window;
X  {
X!   register int w = decode_window (window)->width;
X    /* If this window does not end at the right margin,
X       must deduct one column for the border */
X!   if (w + decode_window (window)->left == screen_width)
X      return w;
X!   return w - 1;
X  }
X
X  DEFUN ("window-hscroll", Fwindow_hscroll, Swindow_hscroll, 0, 1, 0,
X--- 217,229 ----
X    (window)
X       Lisp_Object window;
X  {
X!   register Lisp_Object w = decode_window (window)->width;
X    /* If this window does not end at the right margin,
X       must deduct one column for the border */
X!   if (XINT (w) + XINT(decode_window (window)->left) == screen_width)
X      return w;
X!   XSETINT (w, XINT(w) - 1);
X!   return w;
X  }
X
X  DEFUN ("window-hscroll", Fwindow_hscroll, Swindow_hscroll, 0, 1, 0,
X***************
X*** 246,252 ****
X    if (XFASTINT (ncol) >= (1 << (SHORTBITS - 1)))
X      args_out_of_range (ncol, Qnil);
X    w = decode_window (window);
X!   if (w->hscroll != ncol)
X      clip_changed = 1;         /* Prevent redisplay shortcuts */
X    w->hscroll = ncol;
X    return ncol;
X--- 247,253 ----
X    if (XFASTINT (ncol) >= (1 << (SHORTBITS - 1)))
X      args_out_of_range (ncol, Qnil);
X    w = decode_window (window);
X!   if (XINT(w->hscroll) != XINT(ncol))
X      clip_changed = 1;         /* Prevent redisplay shortcuts */
```

```
X    w->hscroll = ncol;
X    return ncol;
X***************
X*** 334,339 ****
X--- 335,342 ----
X    return pos;
X  }
X

X+ static replace_window (), unshow_buffer ();
X+
X  DEFUN ("delete-window", Fdelete_window, Sdelete_window, 0, 1, "",
X    "Remove WINDOW from the display.  Default is selected window.")
X    (window)
X***************
X*** 1753,1758 ****
X--- 1804,1811 ----
X      count += count_windows (XWINDOW (window->hchild));
X    return count;
X  }
X+
X+ static int save_window_save ();
X
X  DEFUN ("current-window-configuration",
X    Fcurrent_window_configuration, Scurrent_window_configuration, 0, 0, 0,
Xdiff -c original/xdisp.c ./xdisp.c
X*** original/xdisp.c   Mon Jul 17 20:53:38 1989
X--- ./xdisp.c     Sat Jul  8 16:47:01 1989
X***************
X*** 1882,1888 ****
X        }
X      while ((p1 - start + hscroll - (hscroll > 0)) % tab_width);
X    }
X!       else if (c < 0200 && buffer_defaults.ctl_arrow)
X    {
X      if (p1 >= start)
X        *p1 = '^';
X--- 1882,1888 ----
X        }
X      while ((p1 - start + hscroll - (hscroll > 0)) % tab_width);
X    }
X!       else if (c < 0200 && XINT(buffer_defaults.ctl_arrow))
X    {
X      if (p1 >= start)
X        *p1 = '^';
END_OF_FILE
if test 45058 -ne `wc -c <'cdiff-18.52-atari'`; then
    echo shar: \"'cdiff-18.52-atari'\" unpacked with wrong size!
fi
# end of 'cdiff-18.52-atari'
fi
if test -f 's-tos.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'s-tos.h'\"
```

```
else
echo shar: Extracting \"'s-tos.h'\" \(4787 characters\)
sed "s/^X//" >'s-tos.h' <<'END_OF_FILE'
X/* Template for s- header files.
X   This file describes the parameters that s- files should define or not.
X   Copyright (C) 1985 Richard M. Stallman.
X
XThis file is part of GNU Emacs.
X
XGNU Emacs is distributed in the hope that it will be useful,
Xbut WITHOUT ANY WARRANTY.  No author or distributor
Xaccepts responsibility to anyone for the consequences of using it
Xor for whether it serves any particular purpose or works at all,
Xunless he says so in writing.  Refer to the GNU Emacs General Public
XLicense for full details.
X
XEveryone is granted permission to copy, modify and redistribute
XGNU Emacs, but only under the conditions described in the
XGNU Emacs General Public License.   A copy of this license is
Xsupposed to have been given to you along with GNU Emacs so you
Xcan know your rights and responsibilities.  It should be in a
Xfile named COPYING.  Among other things, the copyright notice
Xand this notice must be preserved on all copies.  */
X
X
X/*
X *  Define symbols to identify the version of Unix this is.
X *  Define all the symbols that apply correctly.
X */
X
X/* #define UNIPLUS */
X/* #define USG5 */
X/* #define USG */
X/* #define HPUX */
X/* #define UMAX */
X/* #define BSD4_1 */
X/* #define BSD4_2 */
X/* #define BSD4_3 */
X/* #define BSD */
X/* #define VMS */
X
X/* SYSTEM_TYPE should indicate the kind of system you are using.
X It sets the Lisp variable system-type.  */
X
X#define SYSTEM_TYPE "Atari-ST TOS"
X
X/* NOMULTIPLEJOBS should be defined if your system's shell
X does not have "job control" (the ability to stop a program,
X run some other program, then continue the first one).  */
X
X#define NOMULTIPLEJOBS
X
X/* Emacs can read input using SIGIO and buffering characters itself,
```

```
X   or using CBREAK mode and making C-g cause SIGINT.
X   The choice is controlled by the variable interrupt_input.
X   Define INTERRUPT_INPUT to make interrupt_input = 1 the default (use SIGIO)
X
X   SIGIO can be used only on systems that implement it (4.2 and 4.3).
X   CBREAK mode has two disadvatages
X     1) At least in 4.2, it is impossible to handle the Meta key properly.
X        I hear that in system V this problem does not exist.
X     2) Control-G causes output to be discarded.
X        I do not know whether this can be fixed in system V.
X
X   Another method of doing input is planned but not implemented.
X   It would have Emacs fork off a separate process
X   to read the input and send it to the true Emacs process
X   through a pipe.
X*/
X
X/* #define INTERRUPT_INPUT */
X
X/* Letter to use in finding device name of first pty,
X  if system supports pty's.  'a' means it is /dev/ptya0  */
X
X#define FIRST_PTY_LETTER 'a'
X
X/*
X *  Define HAVE_TIMEVAL if the system supports the BSD style clock values.
X *  Look in <sys/time.h> for a timeval structure.
X */
X
X/* #define HAVE_TIMEVAL */
X
X/*
X *  Define HAVE_SELECT if the system supports the `select' system call.
X */
X
X/* #define HAVE_SELECT */
X
X/*
X *  Define HAVE_PTYS if the system supports pty devices.
X */
X
X/* #define HAVE_PTYS */
X
X/*
X *  Define NONSYSTEM_DIR_LIBRARY to make Emacs emulate
X *      The 4.2 opendir, etc., library functions.
X */
X
X/* #define NONSYSTEM_DIR_LIBRARY */
X
X/* Define this symbol if your system has the functions bcopy, etc. */
X
X#define BSTRING
```

```
X
X/* subprocesses should be defined if you want to
X   have code for asynchronous subprocesses
X   (as used in M-x compile and M-x shell).
X   This is generally OS dependent, and not supported
X   under most USG systems. */
X
X/* #define subprocesses */
X
X/* If your system uses COFF (Common Object File Format) then define the
X   preprocessor symbol "COFF". */
X
X/* #define COFF */
X
X/* define MAIL_USE_FLOCK if the mailer uses flock
X   to interlock access to /usr/spool/mail/$USER.
X   The alternative is that a lock file named
X   /usr/spool/mail/$USER.lock.  */
X
X/* #define MAIL_USE_FLOCK */
X
X/* Define CLASH_DETECTION if you want lock files to be written
X   so that Emacs can tell instantly when you try to modify
X   a file that someone else has modified in his Emacs.  */
X
X/* #define CLASH_DETECTION */
X
X/* Here, on a separate page, add any special hacks needed
X   to make Emacs work on this system.  For example,
X   you might define certain system call names that don't
X   exist on your system, or that do different things on
X   your system and must be used only through an encapsulation
X   (Which you should place, by convention, in sysdep.c).  */
X

X/* Some compilers tend to put everything declared static
X   into the initialized data area, which becomes pure after dumping Emacs.
X   On these systems, you must #define static as nothing to foil this.
X   Note that emacs carefully avoids static vars inside functions.  */
X
X/* #define static */
X
X#define sigsetmask(n)
X#define PENDING_OUTPUT_COUNT(FILE) (((struct file *)FILE)->buf_max - ((struct
file *)FILE)->buf_index)
X
X#define MAINTAIN_ENVIRONMENT
END_OF_FILE
if test 4787 -ne `wc -c <'s-tos.h'`; then
    echo shar: \"'s-tos.h'\" unpacked with wrong size!
fi
# end of 's-tos.h'
fi
```

```
if test -f 'm-atari.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'m-atari.h'\"
else
echo shar: Extracting \"'m-atari.h'\" \(4104 characters\)
sed "s/^X//" >'m-atari.h' <<'END_OF_FILE'
X/* m- file template.
X   Copyright (C) 1985 Richard M. Stallman.
X
XThis file is part of GNU Emacs.
X
XGNU Emacs is distributed in the hope that it will be useful,
Xbut WITHOUT ANY WARRANTY.  No author or distributor
Xaccepts responsibility to anyone for the consequences of using it
Xor for whether it serves any particular purpose or works at all,
Xunless he says so in writing.  Refer to the GNU Emacs General Public
XLicense for full details.
X
XEveryone is granted permission to copy, modify and redistribute
XGNU Emacs, but only under the conditions described in the
XGNU Emacs General Public License.   A copy of this license is
Xsupposed to have been given to you along with GNU Emacs so you
Xcan know your rights and responsibilities.  It should be in a
Xfile named COPYING.  Among other things, the copyright notice
Xand this notice must be preserved on all copies.  */
X
X
X/* The following three symbols give information on
X the size of various data types.  */
X
X#define SHORTBITS 16       /* Number of bits in a short */
X
X#define INTBITS 32         /* Number of bits in an int */
X
X#define LONGBITS 32        /* Number of bits in a long */
X
X/* Define BIG_ENDIAN iff lowest-numbered byte in a word
X   is the most significant byte.  */
X
X#define BIG_ENDIAN
X
X/* Define NO_ARG_ARRAY if you cannot take the address of the first of a
X * group of arguments and treat it as an array of the arguments.  */
X
X/* #define NO_ARG_ARRAY */
X
X/* Define WORD_MACHINE if addresses and such have
X * to be corrected before they can be used as byte counts.  */
X
X/* #define WORD_MACHINE */
X
X/* Define how to take a char and sign-extend into an int.
X   On machines where char is signed, this is a no-op.  */
X
```

```
X#define SIGN_EXTEND_CHAR(c) (c)
X
X/* Now define a symbol for the cpu type, if your compiler
X   does not define it automatically:
X   vax, m68000, ns16000, pyramid, orion, tahoe and APOLLO
X   are the ones defined so far.  */
X
X/* Use type int rather than a union, to represent Lisp_Object */
X/* This is desirable for most machines.  */
X
X#define NO_UNION_TYPE
X
X/* Define EXPLICIT_SIGN_EXTEND if XINT must explicitly sign-extend
X   the 24-bit bit field into an int.  In other words, if bit fields
X   are always unsigned.
X
X   If you use NO_UNION_TYPE, this flag does not matter.  */
X
X#define EXPLICIT_SIGN_EXTEND
X
X/* Data type of load average, as read out of kmem.  */
X
X/* #define LOAD_AVE_TYPE long */
X
X/* Convert that into an integer that is 100 for a load average of 1.0  */
X
X/* #define LOAD_AVE_CVT(x) (int) (((double) (x)) * 100.0 / 100.0) */
X
X/* Define CANNOT_DUMP on machines where unexec does not work.
X   Then the function dump-emacs will not be defined
X   and temacs will do (load "loadup") automatically unless told otherwise.  */
X
X#define CANNOT_DUMP
X#ifdef CANNOT_DUMP
X#define PURESIZE  16384
X#endif
X
X/* Define VIRT_ADDR_VARIES if the virtual addresses of
X   pure and impure space as loaded can vary, and even their
X   relative order cannot be relied on.
X
X   Otherwise Emacs assumes that text space precedes data space,
X   numerically.  */
X
X/* #define VIRT_ADDR_VARIES */
X
X/* Define C_ALLOCA if this machine does not support a true alloca
X   and the one written in C should be used instead.
X   Define HAVE_ALLOCA to say that the system provides a properly
X   working alloca function and it should be used.
X   Define neither one if an assembler-language alloca
X   in the file alloca.s should be used.  */
X
```

```
X/* #define C_ALLOCA */
X#define HAVE_ALLOCA
X#define alloca    __builtin_alloca
X
X/* Define NO_REMAP if memory segmentation makes it not work well
X   to change the boundary between the text section and data section
X   when Emacs is dumped.  If you define this, the preloaded Lisp
X   code will not be sharable; but that's better than failing completely.  */
X
X#define NO_REMAP
X
X/* These values are needed by some files (alloc.c, data.c) even if
X   NO_UNION_TYPE is undefined */
X
X#ifndef NO_UNION_TYPE
X#define VALBITS 24
X#define GCTYPEBITS 7
X#define VALMASK ((1<<VALBITS) - 1)
X#define GCTYPEMASK ((1<<GCTYPEBITS) - 1)
X#define MARKBIT (1 << (VALBITS + GCTYPEBITS))
X#endif
X
X#define SWITCH_ENUM_BUG
X#define abort()   _abort_internal(__LINE__,__FILE__)
END_OF_FILE
if test 4104 -ne `wc -c <'m-atari.h'`; then
    echo shar: \"'m-atari.h'\" unpacked with wrong size!
fi
# end of 'm-atari.h'
fi
echo shar: End of shell archive.
exit 0


From 5806306135345166762
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,12a4bdbeb4ee2e95,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-08-10 12:46:53 PST
Path: utzoo!attcan!uunet!cs.utexas.edu!csd4.milw.wisc.edu!bionet!apple!sun-barr!
newstop!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i070: protect -- Virus killer/protector
Keywords: 68000
Message-ID: <34080@grapevine.uucp>
Date: 10 Aug 89 19:46:53 GMT
Sender: news@grapevine.uucp
Lines: 152
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: f-leoe@ifi.uio.no (Lars-Erik 0sterud)
Posting-number: Volume 2, Issue 70
```

```
Archive-name: protect

Small virus destruction and protection utility...

 PEA intro(PC)    ;start beskjed
 MOVE.W #9,-(SP)  ;print line
 TRAP #1          ;gemdos
 ADDQ.L #6,SP
check_disk:
 MOVE.W #7,-(SP)  ;les tegn
 TRAP #1          ;gemdos
 ADDQ.L #2,SP
 CMP.B #'n',D0    ;er det nei ?
 BNE lets_check   ;nei - sjekke disk
 CLR.W -(SP)      ;exit
 TRAP #1
lets_check:
 BSR read_boot    ;lese disk boot-sector
 BSR checksum     ;checksum i D7
 CMP.W #$1234,D7  ;executable ?
 BNE no_danger    ;nei - ingen fare
 PEA warning(PC)  ;skrive advarsel
 MOVE.W #9,-(SP)  ;print line
 TRAP #1          ;gemdos
 ADDQ.L #6,SP
 BRA immunize     ;kverke og beskytte disk
no_danger:
 LEA buf(PC),A6
 CMP.B #$60,(A6)
 BNE immunize
 PEA immune(PC)   ;skrive beskjed
 MOVE.W #9,-(SP)  ;print line
 TRAP #1          ;gemdos
 ADDQ.L #6,SP
 BRA more_disks   ;flere disketter ?
immunize:
 PEA protect(PC)  ;beskytte disk beskjed
 MOVE.W #9,-(SP)  ;print line
 TRAP #1          ;gemdos
 ADDQ.L #6,SP
 MOVE.W #7,-(SP)  ;les tegn
 TRAP #1          ;gemdos
 ADDQ.L #2,SP
 CMP.B #'y',D0    ;er det ja ?
 BNE more_disks   ;nei - ny disk
 BSR protect_disk ;beskytte disk
more_disks:
 PEA check(PC)    ;flere disker beskjed
 MOVE.W #9,-(SP)  ;print line
 TRAP #1          ;gemdos
 ADDQ.L #6,SP
 BRA check_disk   ;sjekke ny disk ?
```

```
checksum:
 CLR.W D7
 LEA buf(PC),A6
 MOVE.W #255,D6
loop3:
 ADD.W (A6)+,D7
 DBRA D6,loop3
 RTS

read_boot:
 MOVE.W #1,-(sp)     ;Antall sektorer paa stack
 MOVE.W #0,-(sp)     ;Side 0 paa stack
 MOVE.W #0,-(sp)     ;Track 0 paa stack
 MOVE.W #1,-(sp)     ;Sector 1 paa stack
 MOVE.W #0,-(sp)     ;Drive 0 (A) paa stack
 CLR.L -(sp)         ;Ubrukt paa stack
 PEA buf(PC)         ;buffer-adresse paa stack
 MOVE.W #8,-(sp)     ;XBIOS 8 (floprd) paa stack
 TRAP #14            ;Kalle opp XBIOS
 ADD.L #20,sp        ;Rette opp stack-pointer
 RTS

protect_disk:
 LEA buf(PC),A6
 LEA bstart(PC),A5
 MOVE.L (A5),(A6)    ;Start paa bootsector
 MOVE.L 4(A5),4(A6)

 ADD.L #30,A6        ;Ledig plass 30 frem
 LEA btxt(PC),A5     ;Adresse til bootprg i A5
loop:
 MOVE.B (A5)+,(A6)+  ;Flytte fra adresse til adresse
 TST.B (A5)          ;0 betyr slutt
 BNE loop            ;Neste tegn

 MOVE.W #208,D6      ;Blanke ut resten
loop2:               ;av boot-sektoren
 CLR.B (A6)+
 DBRA D6,loop2

 MOVE.W #1,-(sp) ;Antall sektorer paa stack
 MOVE.W #0,-(sp) ;Side 0 paa stack
 MOVE.W #0,-(sp) ;Track 0 paa stack
 MOVE.W #1,-(sp) ;Sector 1 paa stack
 MOVE.W #0,-(sp) ;Drive 0 (A) paa stack
 CLR.L -(sp)     ;Ubrukt paa stack
 PEA buf(PC)     ;buffer-adresse paa stack
 MOVE.W #9,-(sp) ;XBIOS 9 (flopwr) paa stack
 TRAP #14        ;Kalle opp XBIOS
 ADD.L #20,sp    ;Rette opp stack-pointer
 RTS

intro:
```

```
 DC.B 27,'E',10,"  ABK-BBS's Virus Killer & Protector",13,10,10
 DC.B '  ',$BD,' by Lars-Erik 2sterud - July 1989',13,10,10,10
 DC.B '  Call ABK-BBS in Oslo, Norway  +47 2 13 26 59',13,10,10
 DC.B '  Adress:  2kriveien 39, N-1349 Rykkinn, Norway',13,10,10
 DC.B '  E-mail:  leoe',64,'ifi.uio.no / f-leoe',64,'ifi.uio.no'
check:
 DC.B 13,10,10,10,'  Check disk in drive A: for Auto-Booting program ? ',0

warning:
 DC.B 13,10,10,10,7,'  WARNING, Disk is AUTO-',7
 DC.B 'BOOTING ',7,'(could be a VIRUS)',13,7,0
immune:
 DC.B 13,10,10,'  Disk in drive A: is allready protected.',13,0
protect:
 DC.B 13,10,10,'  Do you want to KILL & PROTECT disk in drive A: ? ',0,0

bstart:
 DC.W $601C     ;Hoppe til prg (buf+30)
 DC.B 'Loader'  ;Narre ekte virus

btxt:           ;Legges inn p boot-sektor
 RTS
 DC.B "####################"
 DC.B " Disk protected "
 DC.B "  by ABK-BBS's  "
 DC.B "   Anti-Virus   "
 DC.B "####################"
 DC.B "  Dial ABK-BBS  "
 DC.B "  Oslo, Norway  "
 DC.B " +47 2 13 26 59 "
 DC.B "####################"
 DC.B "  L.E. 0sterud  "
 DC.B "  0kriveien 39  "
 DC.B " N-1349 Rykkinn "
 DC.B "     Norway     "
 DC.B "####################"
 DC.B " Send E-mail to "
 DC.B "leoe",64,"ifi.uio.no "
 DC.B "####################",0,0

buf:  ;Buffer for disk read/write (gidder ikke sette av plassen n)

 END
```


From 5517341731938874434
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,b163a34cf8c83f5f,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-08-10 12:56:04 PST
Path: utzoo!attcan!uunet!cs.utexas.edu!sun-barr!newstop!sun!sunfedcomm!grapevine!
koreth%panarthea.ebay@sun.com

From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i071: ff31 -- File finder version 3.1
Keywords: Pascal
Message-ID: <34084@grapevine.uucp>
Date: 10 Aug 89 19:56:04 GMT
Sender: news@grapevine.uucp
Lines: 247
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: f-leoe@ifi.uio.no (Lars-Erik 0sterud)
Posting-number: Volume 2, Issue 71
Archive-name: ff31

THe newest version of my File-Finder...
Start with no parameters for Help...
Source-code in Personal Pascal 2...

leoe@ifi.uio.no / f-leoe@ifi.uio.no

```
{$C-,D-,E-,P-,R-,T-,S100} {** All sjekking av, Ikke rense hele RAM'en **}

PROGRAM file_finder_3_1;

TYPE fname   = PACKED ARRAY [1..14] OF CHAR;
     filerec = PACKED RECORD
                 reserved : PACKED ARRAY [0..19] OF BYTE;
                 attributes,reserved2: BYTE;
                 date_stamp,time_stamp : INTEGER;
                 file_size : LONG_INTEGER;
                 file_name : fname;
               END; {record filrec}

     path_name = PACKED ARRAY [1..80] OF CHAR;

     directory = ^dirtype;
     dirtype = RECORD
                 dirname:STRING;
                 neste:directory;
               END; {record dirtype}

VAR buffer:filerec;
    drive,path:STRING;
    choice,start,stopp,printer,a:CHAR;
    print_ut:TEXT;
    run_program:BOOLEAN;
    folders,files,found:INTEGER;

PROCEDURE Datestring(date:INTEGER);
VAR dag,mnd,aar:INTEGER;
BEGIN
  dag:=date & 31;
```

```
  mnd:=ShR(date,5) & 15;
  aar:=1980+ShR(date,9) & 127;
  WRITE(print_ut,'  ');
  IF dag<10 THEN WRITE(print_ut,'0');
  WRITE(print_ut,dag,'/');
  IF mnd<10 THEN WRITE(print_ut,'0');
  WRITELN(print_ut,mnd,'-',aar);
END; {proc datestring}

PROCEDURE wait_for_key;
GEMDOS($07);

PROCEDURE Set_Dta    (VAR buffer:filerec);
GEMDOS($1A);         {*** Set Disk Transfer-buffer ***}

FUNCTION  Get_First  (VAR path:path_name;attributes:INTEGER):BOOLEAN;
GEMDOS($4E);         {*** Find first matching file ***}

FUNCTION  Get_Next:BOOLEAN;
GEMDOS($4F);         {*** Find next match ***}

PROCEDURE make_array(innavn:STRING;VAR utnavn:path_name);
VAR a:INTEGER;
BEGIN
  FOR a:=1 TO Length(innavn) DO utnavn[a]:=innavn[a];
  utnavn[a]:=CHR(0);  {*** Slutt paa tekststrengen ***}
END; {proc make_array}

PROCEDURE make_string(innavn:fname;VAR utnavn:STRING);
VAR a:INTEGER;
BEGIN
  a:=1;
  WHILE innavn[a]<>CHR(0) DO a:=a+1;
  utnavn:=Copy(innavn,1,a-1);
END; {func wrt_name}

PROCEDURE search(name,path:STRING);
VAR temp:STRING;
    funnet:BOOLEAN;
    sdirpath,sfilpath:path_name;
    start,current,last:directory;
BEGIN
  temp:=Concat(name,'*.*');
  make_array(temp,sdirpath);
  temp:=Concat(name,path);
  make_array(temp,sfilpath);

  WRITE(CHR(13),'  Searching   ',name,path,CHR(27),'K');

  {*** Scan for directories ***}

  NEW(start);
  start^.neste:=NIL;
```

```
  current:=start;
  funnet:=NOT Get_First(sdirpath,16);
  WHILE funnet DO BEGIN
      IF (buffer.attributes=16) THEN BEGIN
         IF (buffer.file_name[1]<>'.') THEN BEGIN
            folders:=folders+1; {*** Telle directorier ***}
            last:=current;
            NEW(current);
            last^.neste:=current;
            make_string(buffer.file_name,temp);
            current^.dirname:=Concat(name,temp,'\');
            current^.neste:=NIL;
         END; {if buffer.attr}
         END
      ELSE files:=files+1; {*** Telle vanlige filer ***}
      funnet:=NOT Get_Next;
  END; {while funnet}

  {*** Scan for file ***}

  funnet:=NOT Get_First(sfilpath,15);
  WHILE funnet DO BEGIN
      found:=found+1; {*** telle antall funnet ***}
      make_string(buffer.file_name,temp);
      temp:=Concat(name,temp);
      WRITE(print_ut,CHR(13),'  Found file  ',temp);
      WRITE(print_ut,buffer.file_size:(52-Length(temp)));
      Datestring(buffer.date_stamp);
      IF run_program THEN
         IF (Pos('.TOS',temp)>0) OR (Pos('.TTP',temp)>0) OR
            (Pos('.PRG',temp)>0) OR (Pos('.APP',temp)>0) THEN BEGIN
            WRITE(CHR(27),'e');CHAIN(temp);WRITELN(CHR(27),'f');
         END; {if executable}
      funnet:=NOT Get_Next;
  END; {while funnet}

  {*** Search next directory - Recursive ! ***}

  WHILE start^.neste<>NIL DO BEGIN
      current:=start^.neste;
      search(current^.dirname,path);
      start^.neste:=current^.neste;
      DISPOSE(current);
  END; {while start^.neste}

  DISPOSE(start);
END; {proc search}

PROCEDURE upcase(VAR tekst:STRING);
VAR a:INTEGER;
BEGIN
  FOR a:=1 TO Length(tekst) DO
      IF tekst[a] IN ['a'..'z'] THEN tekst[a]:=CHR(ORD(tekst[a])-32);
```

```
END; {proc upcase}

FUNCTION peek_l(adresse:LONG_INTEGER):LONG_INTEGER;
VAR magic: RECORD CASE BOOLEAN OF
              FALSE:(long:LONG_INTEGER);
              TRUE :(ptr :^LONG_INTEGER)
           END; {record}
BEGIN
  magic.long:=adresse;
  peek_l:=magic.ptr^
END; {func peek_l}

FUNCTION Super(inn:LONG_INTEGER):LONG_INTEGER;
GEMDOS($20);

FUNCTION disk(drive:CHAR):BOOLEAN;
VAR stack:LONG_INTEGER;
BEGIN
   stack:=Super(0);
   disk:=((ShR(peek_l($4C2),ORD(drive)-65)&1)=1);
   stack:=Super(stack);
END; {func disk}

PROCEDURE get_drive_path(VAR drive:CHAR;VAR path:STRING);
{*** Leser inn drive og sokepath ***}
VAR output,dummy:STRING;
    a:INTEGER;
BEGIN
  IF Cmd_Args<1 THEN BEGIN {*** Ingen parametere ***}
     drive:='0';
     WRITELN('  Use:      FF <name> (<option>) (<output>) (<exec>)',CHR(10));
     WRITELN('  Options:  /H  = Search All Hard-disks (default)');
     WRITELN('            /A  = Search All drives in system');
     WRITELN('            /Sx = Serach Only on Drive x:',CHR(10));
     WRITELN('  Output:   >CON:     = To Screen (default)');
     WRITELN('            >LST:     = To Printer');
     WRITELN('            >FILE.EXT = To Textfile',CHR(10));
     WRITELN('  Execute:  ## = Start programs when found',CHR(10));
     END
  ELSE BEGIN
     Cmd_GetArg(1,path);upcase(path);
     drive:='1';output:='CON:';run_program:=FALSE;
     FOR a:=2 TO Cmd_Args DO BEGIN
         Cmd_GetArg(a,dummy);upcase(dummy);
         IF dummy[1]='/' THEN BEGIN
            CASE dummy[2] OF
               'A' :drive:='2';      {*** Alle driver   ***}
               'S' :drive:=dummy[3]; {*** en drive ***}
               ELSE:drive:='1';      {*** Alle harddisker ***}
            END; {case switch}
            END
         ELSE IF dummy[1]='>' THEN output:=Copy(dummy,2,Length(dummy)-1)
         ELSE IF dummy='##' THEN run_program:=TRUE;
```

```
    END; {for a}
    IF Length(output)<3 THEN REWRITE(print_ut,'CON:')
    ELSE REWRITE(print_ut,output)
  END; {if cmds}
END; {proc get_drive_path}

BEGIN
  files:=0;
  folders:=0;
  found:=0;
  Set_Dta(buffer);
  WRITELN(CHR(27),'f');
  WRITELN('  File-Finder 3.1  ',CHR($BD),' Lars-Erik 0sterud  1989',CHR(10));
  get_drive_path(choice,path);
  IF choice<>'0' THEN BEGIN
     WRITE(print_ut,'  Search  ');
     IF choice='1' THEN BEGIN
       WRITE(print_ut,'All Hard-Disks');
       start:='C';stopp:='Z';
       END
     ELSE IF choice='2' THEN BEGIN
       WRITE(print_ut,'All drives');
       start:='A';stopp:='Z';
       END
     ELSE BEGIN
       WRITE(print_ut,'drive ',choice,':');
       start:=choice;stopp:=choice;
     END; {if choice}
     WRITE(print_ut,'  for  ',path);
     IF run_program THEN WRITE(print_ut,'  and execute programs');
     WRITELN(print_ut,CHR(10));
     FOR a:=start TO stopp DO IF disk(a) THEN BEGIN
         drive:=Concat(a,':\');
         search(drive,path);
     END; {for a}
     WRITE(CHR(27),'l',CHR(7),CHR(13)); {*** Fjerne siste linje ***}
     WRITELN(print_ut);
     WRITE(print_ut,'  ',files,' files, ',folders,' directories searched  -');
     WRITELN(print_ut,'  ',found,' matching files found',CHR(10));
     CLOSE(print_ut);
     WRITE('  Search completed  -');
  END; {for/if}
  WRITE('  Press any key to exit File-Finder');
  wait_for_key;WRITELN;WRITELN;
END.


From -3400124901588621911
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,43118313282ff87d,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-08-11 07:30:30 PST
Path: gmdzi!unido!mcvax!uunet!tut.cis.ohio-state.edu!ucbvax!agate!apple!sun-barr!
```

newstop!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i071: ff31 -- Alternate file-selector version 3.1
Keywords: Pascal
Message-ID: <34081@grapevine.uucp>
Date: 10 Aug 89 19:49:35 GMT
Sender: news@grapevine.uucp
Lines: 247
Approved: koreth%panarthea.ebay@sun.com
Posted: Thu Aug 10 20:49:35 1989

Submitted-by: f-leoe@ifi.uio.no (Lars-Erik 0sterud)
Posting-number: Volume 2, Issue 71
Archive-name: ff31

THe newest version of my File-Finder...
Start with no parameters for Help...
Source-code in Personal Pascal 2...

leoe@ifi.uio.no / f-leoe@ifi.uio.no

```
{$C-,D-,E-,P-,R-,T-,S100} {** All sjekking av, Ikke rense hele RAM'en **}

PROGRAM file_finder_3_1;

TYPE fname   = PACKED ARRAY [1..14] OF CHAR;
     filerec = PACKED RECORD
                 reserved : PACKED ARRAY [0..19] OF BYTE;
                 attributes,reserved2: BYTE;
                 date_stamp,time_stamp : INTEGER;
                 file_size : LONG_INTEGER;
                 file_name : fname;
               END; {record filrec}

     path_name = PACKED ARRAY [1..80] OF CHAR;

     directory = ^dirtype;
     dirtype = RECORD
                 dirname:STRING;
                 neste:directory;
               END; {record dirtype}

VAR buffer:filerec;
    drive,path:STRING;
    choice,start,stopp,printer,a:CHAR;
    print_ut:TEXT;
    run_program:BOOLEAN;
    folders,files,found:INTEGER;

PROCEDURE Datestring(date:INTEGER);
VAR dag,mnd,aar:INTEGER;
```

```
BEGIN
  dag:=date & 31;
  mnd:=ShR(date,5) & 15;
  aar:=1980+ShR(date,9) & 127;
  WRITE(print_ut,'  ');
  IF dag<10 THEN WRITE(print_ut,'0');
  WRITE(print_ut,dag,'/');
  IF mnd<10 THEN WRITE(print_ut,'0');
  WRITELN(print_ut,mnd,'-',aar);
END; {proc datestring}

PROCEDURE wait_for_key;
GEMDOS($07);

PROCEDURE Set_Dta    (VAR buffer:filerec);
GEMDOS($1A);         {*** Set Disk Transfer-buffer ***}

FUNCTION  Get_First  (VAR path:path_name;attributes:INTEGER):BOOLEAN;
GEMDOS($4E);         {*** Find first matching file ***}

FUNCTION  Get_Next:BOOLEAN;
GEMDOS($4F);         {*** Find next match ***}

PROCEDURE make_array(innavn:STRING;VAR utnavn:path_name);
VAR a:INTEGER;
BEGIN
  FOR a:=1 TO Length(innavn) DO utnavn[a]:=innavn[a];
  utnavn[a]:=CHR(0);  {*** Slutt paa tekststrengen ***}
END; {proc make_array}

PROCEDURE make_string(innavn:fname;VAR utnavn:STRING);
VAR a:INTEGER;
BEGIN
  a:=1;
  WHILE innavn[a]<>CHR(0) DO a:=a+1;
  utnavn:=Copy(innavn,1,a-1);
END; {func wrt_name}

PROCEDURE search(name,path:STRING);
VAR temp:STRING;
    funnet:BOOLEAN;
    sdirpath,sfilpath:path_name;
    start,current,last:directory;
BEGIN
  temp:=Concat(name,'*.*');
  make_array(temp,sdirpath);
  temp:=Concat(name,path);
  make_array(temp,sfilpath);

  WRITE(CHR(13),'  Searching   ',name,path,CHR(27),'K');

  {*** Scan for directories ***}
```

```
  NEW(start);
  start^.neste:=NIL;
  current:=start;
  funnet:=NOT Get_First(sdirpath,16);
  WHILE funnet DO BEGIN
      IF (buffer.attributes=16) THEN BEGIN
         IF (buffer.file_name[1]<>'.') THEN BEGIN
            folders:=folders+1; {*** Telle directorier ***}
            last:=current;
            NEW(current);
            last^.neste:=current;
            make_string(buffer.file_name,temp);
            current^.dirname:=Concat(name,temp,'\');
            current^.neste:=NIL;
         END; {if buffer.attr}
         END
      ELSE files:=files+1; {*** Telle vanlige filer ***}
      funnet:=NOT Get_Next;
  END; {while funnet}

  {*** Scan for file ***}

  funnet:=NOT Get_First(sfilpath,15);
  WHILE funnet DO BEGIN
      found:=found+1; {*** telle antall funnet ***}
      make_string(buffer.file_name,temp);
      temp:=Concat(name,temp);
      WRITE(print_ut,CHR(13),'  Found file  ',temp);
      WRITE(print_ut,buffer.file_size:(52-Length(temp)));
      Datestring(buffer.date_stamp);
      IF run_program THEN
         IF (Pos('.TOS',temp)>0) OR (Pos('.TTP',temp)>0) OR
            (Pos('.PRG',temp)>0) OR (Pos('.APP',temp)>0) THEN BEGIN
            WRITE(CHR(27),'e');CHAIN(temp);WRITELN(CHR(27),'f');
         END; {if executable}
      funnet:=NOT Get_Next;
  END; {while funnet}

  {*** Search next directory - Recursive ! ***}

  WHILE start^.neste<>NIL DO BEGIN
      current:=start^.neste;
      search(current^.dirname,path);
      start^.neste:=current^.neste;
      DISPOSE(current);
  END; {while start^.neste}

  DISPOSE(start);
END; {proc search}

PROCEDURE upcase(VAR tekst:STRING);
VAR a:INTEGER;
BEGIN
```

```
  FOR a:=1 TO Length(tekst) DO
      IF tekst[a] IN ['a'..'z'] THEN tekst[a]:=CHR(ORD(tekst[a])-32);
END; {proc upcase}

FUNCTION peek_l(adresse:LONG_INTEGER):LONG_INTEGER;
VAR magic: RECORD CASE BOOLEAN OF
              FALSE:(long:LONG_INTEGER);
              TRUE :(ptr :^LONG_INTEGER)
           END; {record}
BEGIN
  magic.long:=adresse;
  peek_l:=magic.ptr^
END; {func peek_l}

FUNCTION Super(inn:LONG_INTEGER):LONG_INTEGER;
GEMDOS($20);

FUNCTION disk(drive:CHAR):BOOLEAN;
VAR stack:LONG_INTEGER;
BEGIN
   stack:=Super(0);
   disk:=((ShR(peek_l($4C2),ORD(drive)-65)&1)=1);
   stack:=Super(stack);
END; {func disk}

PROCEDURE get_drive_path(VAR drive:CHAR;VAR path:STRING);
{*** Leser inn drive og sokepath ***}
VAR output,dummy:STRING;
    a:INTEGER;
BEGIN
  IF Cmd_Args<1 THEN BEGIN {*** Ingen parametere ***}
     drive:='0';
     WRITELN('  Use:      FF <name> (<option>) (<output>) (<exec>)',CHR(10));
     WRITELN('  Options:  /H  = Search All Hard-disks (default)');
     WRITELN('            /A  = Search All drives in system');
     WRITELN('            /Sx = Serach Only on Drive x:',CHR(10));
     WRITELN('  Output:   >CON:     = To Screen (default)');
     WRITELN('            >LST:     = To Printer');
     WRITELN('            >FILE.EXT = To Textfile',CHR(10));
     WRITELN('  Execute:  ## = Start programs when found',CHR(10));
     END
  ELSE BEGIN
     Cmd_GetArg(1,path);upcase(path);
     drive:='1';output:='CON:';run_program:=FALSE;
     FOR a:=2 TO Cmd_Args DO BEGIN
         Cmd_GetArg(a,dummy);upcase(dummy);
         IF dummy[1]='/' THEN BEGIN
            CASE dummy[2] OF
               'A' :drive:='2';      {*** Alle driver   ***}
               'S' :drive:=dummy[3]; {*** en drive ***}
               ELSE:drive:='1';      {*** Alle harddisker ***}
            END; {case switch}
            END
```

```
          ELSE IF dummy[1]='>' THEN output:=Copy(dummy,2,Length(dummy)-1)
          ELSE IF dummy='##' THEN run_program:=TRUE;
      END; {for a}
      IF Length(output)<3 THEN REWRITE(print_ut,'CON:')
      ELSE REWRITE(print_ut,output)
  END; {if cmds}
END; {proc get_drive_path}

BEGIN
  files:=0;
  folders:=0;
  found:=0;
  Set_Dta(buffer);
  WRITELN(CHR(27),'f');
  WRITELN('  File-Finder 3.1  ',CHR($BD),' Lars-Erik 0sterud  1989',CHR(10));
  get_drive_path(choice,path);
  IF choice<>'0' THEN BEGIN
     WRITE(print_ut,'  Search  ');
     IF choice='1' THEN BEGIN
       WRITE(print_ut,'All Hard-Disks');
       start:='C';stopp:='Z';
       END
     ELSE IF choice='2' THEN BEGIN
       WRITE(print_ut,'All drives');
       start:='A';stopp:='Z';
       END
     ELSE BEGIN
       WRITE(print_ut,'drive ',choice,':');
       start:=choice;stopp:=choice;
     END; {if choice}
     WRITE(print_ut,'  for  ',path);
     IF run_program THEN WRITE(print_ut,'  and execute programs');
     WRITELN(print_ut,CHR(10));
     FOR a:=start TO stopp DO IF disk(a) THEN BEGIN
         drive:=Concat(a,':\');
         search(drive,path);
     END; {for a}
     WRITE(CHR(27),'l',CHR(7),CHR(13)); {*** Fjerne siste linje ***}
     WRITELN(print_ut);
     WRITE(print_ut,'  ',files,' files, ',folders,' directories searched  -');
     WRITELN(print_ut,'  ',found,' matching files found',CHR(10));
     CLOSE(print_ut);
     WRITE('  Search completed  -');
  END; {for/if}
  WRITE('  Press any key to exit File-Finder');
  wait_for_key;WRITELN;WRITELN;
END.


From 4160933185904754632
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,f9d519e2c3e18777,start
X-Google-Attributes: gid100493,public
```

X-Google-ArrivalTime: 1989-08-19 10:53:04 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!csd4.csd.uwm.edu!cs.utexas.edu!
sun-barr!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i072: gcclib2 -- Patches to GCC library part01/07
Keywords: shar, C
Message-ID: <34134@grapevine.uucp>
Date: 19 Aug 89 17:53:04 GMT
Sender: news@grapevine.uucp
Lines: 1619
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: uunet.UU.NET!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 2, Issue 72
Archive-name: gcclib2/part01

This is my version of the gcc library. It was used to port GNU Emacs and many
other unix programs to the st. See the README for more details.
- Edgar

```
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of archive 1 (of 7)."
# Contents:  MANIFEST README abort.c access.c.cdiff atol.c calloc.c
#   chdir.c chmod.c creat.c ctype.c.cdiff env.c exit.c fcntl.c
#   fdopen.c.cdiff fgets.c.cdiff fhandle.c files.c.cdiff fprintf.c
#   fputc.c.cdiff freopen.c.cdiff ftell.c getpage.c getpid.c include
#   include/assert.h include/param.h include/sgtty.h include/stdarg.h
#   include/sys include/sys/dir.h include/sys/errno.h
#   include/sys/file.h include/sys/ioctl.h include/sys/param.h
#   include/sys/signal.h include/sys/stat.h include/sys/time.h
#   include/sys/times.h include/sys/types.h include/time.h
#   include/times.h include/types.h link.c lseek.c mkdir.c parse-op.c
#   perror.c rand.c random.c readdir.c.cdiff rename.c scanf.c.cdiff
#   signame.c strcpy.c.cdiff strlen.c.cdiff times.c uid.c unlink.c
#   utime.c vsprintf.c
# Wrapped by roeder@sbsvax on Wed Aug 16 22:02:54 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'MANIFEST' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'MANIFEST'\"
else
echo shar: Extracting \"'MANIFEST'\" \(3555 characters\)
sed "s/^X//" >'MANIFEST' <<'END_OF_FILE'
X   File Name      Archive #Description
X-----------------------------------------------------------
X MANIFEST                   1   This shipping list
```

```
X README                    1
X abort.c                   1
X access.c.cdiff            1
X alrm_nas.s                3
X atexit.c                  2
X atol.c                    1
X calloc.c                  1
X chdir.c                   1
X chmod.c                   1
X close.c                   2
X closedir.c.cdiff          2
X console.c.cdiff           2
X convert.c                 2
X creat.c                   1
X crt0.s                    5
X ctime.c                   2
X ctype.c.cdiff             1
X env.c                     1
X exit.c                    1
X fclose.c                  2
X fcntl.c                   1
X fdopen.c.cdiff            1
X fflush.c                  2
X fgets.c.cdiff             1
X fhandle.c                 1
X files.c.cdiff             1
X fopen.c                   2
X fprintf.c                 1
X fputc.c.cdiff             1
X freopen.c.cdiff           1
X ftell.c                   1
X getpage.c                 1
X getpid.c                  1
X getwd.c                   2
X include                   1
X include/assert.h          1
X include/ctype.h           2
X include/dir.h             2
X include/errno.h           3
X include/fcntl.h           2
X include/file.h            2
X include/ioctl.h           2
X include/osbind.h          6
X include/param.h           1
X include/sgtty.h           1
X include/siglist.h         2
X include/signal.h          3
X include/stat.h            3
X include/stdarg.h          1
X include/stdio.h           3
X include/stdlib.h          3
X include/sys               1
X include/sys/dir.h         1
```

```
X include/sys/errno.h        1
X include/sys/file.h         1
X include/sys/ioctl.h        1
X include/sys/param.h        1
X include/sys/signal.h       1
X include/sys/stat.h         1
X include/sys/time.h         1
X include/sys/times.h        1
X include/sys/types.h        1
X include/sysvars.h          2
X include/tchars.h           2
X include/time.h             1
X include/times.h            1
X include/types.h            1
X ioctl.c.cdiff              3
X isatty.c.cdiff             2
X link.c                     1
X localtim.c                 2
X lseek.c                    1
X malloc.c                   7
X memory.c.cdiff             4
X mkdir.c                    1
X open.c                     3
X opendir.c.cdiff            3
X parse-op.c                 1
X pause.c                    2
X perror.c                   1
X pexecv.c                   4
X rand.c                     1
X random.c                   1
X read.c                     5
X readdir.c.cdiff            1
X rename.c                   1
X scanf.c.cdiff              1
X sfguts.c.cdiff             2
X signal.c                   4
X signame.c                  1
X st-error.c                 3
X stat.c                     4
X std-guts.h                 2
X std-init.c                 3
X strcpy.c.cdiff             1
X strlen.c.cdiff             1
X strstr.c                   2
X system.c                   4
X tchars.c                   2
X time.c                     3
X timer.c                    4
X times.c                    1
X trap-1.c                   3
X trap-13.c                  3
X trap-14.c                  2
X uid.c                      1
```

```
X unlink.c                   1
X utime.c                    1
X vsprintf.c                 1
X write.c.cdiff              3
END_OF_FILE
if test 3555 -ne `wc -c <'MANIFEST'`; then
    echo shar: \"'MANIFEST'\" unpacked with wrong size!
fi
# end of 'MANIFEST'
fi
if test -f 'README' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'README'\"
else
echo shar: Extracting \"'README'\" \(2275 characters\)
sed "s/^X//" >'README' <<'END_OF_FILE'
XThis is my version of the GNU C compiler library. I used it to port GNU Emacs
Xto the Atari ST. I have included the original files when the diffs would be
Xlarger. Context diffs are produced against the current version at the
Xterminator archive server. Also included are some of the include files which
Xhave changed.  The changes are:
X    - Filenames are converted from unix to tos syntax and vice versa in
X      all file handling routines. So you need no longer worry about the
X      use of slashes in filenames when porting programs from unix. The
X      old writing with backslash is also supported.
X    - The time function now returns seconds since Jan 1 1970. All time
X      functions have changed to deal with this format.
X    - Error numbers are now positive (according to the ANSI standard).
X    - stat/fstat now get similar information than their unix counterparts.
X    - O_APPEND is now handled correctly.
X    - Some error checks have been included in string functions.
X    - crt0.s uses xArg parameter passing (also provided by Pexecv).
X    - Some routines now return more reasonable values than before (getpid).
X    - read recognizes Alternate as Meta key and gives escape sequences for
X      the function keys. O_NDELAY mode is now implemented.
X    - The signal stuff was almost completly rewritten. More signals are
X      now possible. SIGINT as generated by tos can be ignored. SIGTSTP
X      gives you an interactive shell. SIGALRM and SIGHUP are served even in
X      the middle of an operating system call.
X    - The bindings in osbind.h have been rewritten to inline functions. Now
X      they are faster and (for Super()) more correct.
X    - Resident shells at _shell_p are recognized and can be accessed with
X      three new system calls.
X    - Information about the running time can be obtained from clock() or
X      times().
X    - atexit() is now supported.
XWhenever possible i tried to make my routines conforming to the new ANSI C
Xstandard or the corresponding unix-library functions. Because porting is easier
Xif you have a good library, if i had to choose between a change in a program
Xor a new library routine, i most of the time took the latter decision.
XSure there would be only some small changes for this particular program.
X
X- Edgar
X
```

```
X                  If something can be done automatically,
X                  the computer should do the work for you.
END_OF_FILE
if test 2275 -ne `wc -c <'README'`; then
    echo shar: \"'README'\" unpacked with wrong size!
fi
# end of 'README'
fi
if test -f 'abort.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'abort.c'\"
else
echo shar: Extracting \"'abort.c'\" \(546 characters\)
sed "s/^X//" >'abort.c' <<'END_OF_FILE'
X#include <signal.h>
X#include <stdio.h>
X
Xabort()
X{
X  int * stackp;
X  int i;
X
X  stackp = (int * )&stackp;
X  fprintf(stderr, "\r\n\n!!! Aborting!!! stack = \r\n");
X  for(i = 0 ; i < 16 ; i++)
X    fprintf(stderr,"%lx: %lx\r\n", stackp, *stackp++);
X  getchar();
X  raise(SIGABRT);
X  exit(-999);
X}
X
X_abort_internal(line, file)
Xint line;
Xchar * file;
X{
X  fprintf(stderr, "Aborting at line %d of file %s\r\n", line, file);
X  abort();        /* note that this won't work of compile this
X                with the define that got us to this fun in
X                the first place */
X}
END_OF_FILE
if test 546 -ne `wc -c <'abort.c'`; then
    echo shar: \"'abort.c'\" unpacked with wrong size!
fi
# end of 'abort.c'
fi
if test -f 'access.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'access.c.cdiff'\"
else
echo shar: Extracting \"'access.c.cdiff'\" \(405 characters\)
sed "s/^X//" >'access.c.cdiff' <<'END_OF_FILE'
X*** terminator/access.c    Fri Aug 11 18:51:39 1989
X--- new-lib/access.c   Thu Jul  6 16:51:42 1989
X***************
```

```
X*** 6,7 ****
X--- 6,9 ----
X   *
X+  * er  : added user/group/other distinction
X+  *
X   * $Log:   access.c,v $
X***************
X*** 24,26 ****
X         return -1;
X!   if ((statb.st_mode & mode) == mode)
X         return 0;
X--- 26,28 ----
X         return -1;
X!   if (((statb.st_mode >> 6) & mode) == mode)
X         return 0;
END_OF_FILE
if test 405 -ne `wc -c <'access.c.cdiff'`; then
    echo shar: \"'access.c.cdiff'\" unpacked with wrong size!
fi
# end of 'access.c.cdiff'
fi
if test -f 'atol.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'atol.c'\"
else
echo shar: Extracting \"'atol.c'\" \(51 characters\)
sed "s/^X//" >'atol.c' <<'END_OF_FILE'
X
Xlong atol(char *num) { return((long)atoi(num)); }
END_OF_FILE
if test 51 -ne `wc -c <'atol.c'`; then
    echo shar: \"'atol.c'\" unpacked with wrong size!
fi
# end of 'atol.c'
fi
if test -f 'calloc.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'calloc.c'\"
else
echo shar: Extracting \"'calloc.c'\" \(229 characters\)
sed "s/^X//" >'calloc.c' <<'END_OF_FILE'
X#define NULL (char *) 0
X
Xextern char  *malloc();
X
Xchar * calloc(n, sz)
Xunsigned n, sz;
X{
X  char *r, *s;
X  unsigned total;
X
X  total = n * sz;
X  if ((r = s = malloc(total)) != NULL)
X    {
X    while (total--)
```

```
X         *s++ = 0;
X    }
X  return(r);
X}
END_OF_FILE
if test 229 -ne `wc -c <'calloc.c'`; then
    echo shar: \"'calloc.c'\" unpacked with wrong size!
fi
# end of 'calloc.c'
fi
if test -f 'chdir.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'chdir.c'\"
else
echo shar: Extracting \"'chdir.c'\" \(129 characters\)
sed "s/^X//" >'chdir.c' <<'END_OF_FILE'
X#include <osbind.h>
X
Xint chdir(char *d)
X{
X    char dir[strlen(d)+1];
X
X    strcpy(dir,d);
X    unix_to_tos(dir);
X    return(Dsetpath(dir));
X}
X
END_OF_FILE
if test 129 -ne `wc -c <'chdir.c'`; then
    echo shar: \"'chdir.c'\" unpacked with wrong size!
fi
# end of 'chdir.c'
fi
if test -f 'chmod.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'chmod.c'\"
else
echo shar: Extracting \"'chmod.c'\" \(332 characters\)
sed "s/^X//" >'chmod.c' <<'END_OF_FILE'
X#include <types.h>
X#include <osbind.h>
X#include <stat.h>
X
Xint
Xchmod(char *f, int filemode)
X{
X    char filename[strlen(f)+1];
X    int  ret;
X    int  attr;
X
X    strcpy(filename,f);
X    unix_to_tos(filename);
X    attr = 0;
X    if(!(filemode & S_IWRITE)) attr |= FA_RDONLY;
X    if(!(filemode & S_IREAD)) attr |= FA_HIDDEN;
```

```
X    return(Fattrib(filename, 1, attr));
X}
END_OF_FILE
if test 332 -ne `wc -c <'chmod.c'`; then
    echo shar: \"'chmod.c'\" unpacked with wrong size!
fi
# end of 'chmod.c'
fi
if test -f 'creat.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'creat.c'\"
else
echo shar: Extracting \"'creat.c'\" \(114 characters\)
sed "s/^X//" >'creat.c' <<'END_OF_FILE'
X#include <file.h>
X
Xint
Xcreat(char *filename, int mode)
X{
X    return(open(filename,O_WRONLY|O_CREAT|O_TRUNC,mode));
X}
END_OF_FILE
if test 114 -ne `wc -c <'creat.c'`; then
    echo shar: \"'creat.c'\" unpacked with wrong size!
fi
# end of 'creat.c'
fi
if test -f 'ctype.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'ctype.c.cdiff'\"
else
echo shar: Extracting \"'ctype.c.cdiff'\" \(332 characters\)
sed "s/^X//" >'ctype.c.cdiff' <<'END_OF_FILE'
X*** terminator/ctype.c Fri Aug 11 18:51:57 1989
X--- new-lib/ctype.c    Mon Feb  6 18:19:30 1989
X***************
X*** 37,39 ****
X    _L|_P,   _L|_P,   _L|_P,   _P,           /* 'x'  .. '{'  */
X!   _P,  _P,  _P,  _P            /* '|'  .. #x7F  */
X    };
X--- 37,39 ----
X    _L|_P,   _L|_P,   _L|_P,   _P,           /* 'x'  .. '{'  */
X!   _P,  _P,  _P,  _P|_C             /* '|'  .. #x7F  */
X    };
END_OF_FILE
if test 332 -ne `wc -c <'ctype.c.cdiff'`; then
    echo shar: \"'ctype.c.cdiff'\" unpacked with wrong size!
fi
# end of 'ctype.c.cdiff'
fi
if test -f 'env.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'env.c'\"
else
echo shar: Extracting \"'env.c'\" \(318 characters\)
sed "s/^X//" >'env.c' <<'END_OF_FILE'
```

```
X/* stuff for reading env vars sent in by Gulam */
X
Xextern char ** environ;
X
Xchar * getenv (tag)
Xchar * tag;
X{
X  char ** var;
X  char * name;
X  int   len;
X
X  for (var = environ ; name = *var ; var++)
X    {
X    if ((!strncmp(name, tag, len = strlen(tag)))
X        && (name[len] == '='))
X         return(name + len + 1);
X    }
X  return(0);
X}
END_OF_FILE
if test 318 -ne `wc -c <'env.c'`; then
    echo shar: \"'env.c'\" unpacked with wrong size!
fi
# end of 'env.c'
fi
if test -f 'exit.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'exit.c'\"
else
echo shar: Extracting \"'exit.c'\" \(449 characters\)
sed "s/^X//" >'exit.c' <<'END_OF_FILE'
X#include <stdio.h>
X
Xstruct func_list {
X    void          (*func)(void);
X    struct func_list   *next_entry;
X} *_atexit_list = (struct func_list *) 0;
X
Xvoid
Xexit(value)
Xint value;
X{
X  void  (*f)(void);
X
X  while(_atexit_list) {
X    f = _atexit_list->func;
X    /* free(_atexit_list) not necessary, because we _exit anyway */
X    _atexit_list = _atexit_list->next_entry;
X    (*f)();
X  }
X  _fflush_all_files();
X  fflush(stdin);
X  fflush(stdout);
X  fflush(stderr);
```

```
X  _exit(value);
X}
END_OF_FILE
if test 449 -ne `wc -c <'exit.c'`; then
    echo shar: \"'exit.c'\" unpacked with wrong size!
fi
# end of 'exit.c'
fi
if test -f 'fcntl.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'fcntl.c'\"
else
echo shar: Extracting \"'fcntl.c'\" \(327 characters\)
sed "s/^X//" >'fcntl.c' <<'END_OF_FILE'
X#include <file.h>
X#include <fcntl.h>
X#include <errno.h>
X
Xint
Xfcntl(int fd, int func, int flags)
X{
X    int  ret;
X
X    if(fd >= N_HANDLES) {
X        errno = EBADF;
X        return -1;
X    }
X    ret = __handle_stat[fd] & O_NDELAY;
X    if(func == F_SETFL) {
X        flags |= __handle_stat[fd] & (FH_ISATTY|FH_ISAFILE);
X        __handle_stat[fd] = flags;
X    }
X    return ret;
X}
END_OF_FILE
if test 327 -ne `wc -c <'fcntl.c'`; then
    echo shar: \"'fcntl.c'\" unpacked with wrong size!
fi
# end of 'fcntl.c'
fi
if test -f 'fdopen.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'fdopen.c.cdiff'\"
else
echo shar: Extracting \"'fdopen.c.cdiff'\" \(435 characters\)
sed "s/^X//" >'fdopen.c.cdiff' <<'END_OF_FILE'
X*** terminator/fdopen.c    Fri Aug 11 18:52:05 1989
X--- new-lib/fdopen.c   Tue Aug 15 16:30:22 1989
X***************
X*** 1,2 ****
X-
X  /* fdopen */
X--- 1 ----
X***************
X*** 18,20 ****
```

```
X    f->flags = _parse_open_options(mode);
X!   f->file_position = 0;
X  /* should really check first... */
X--- 17,19 ----
X    f->flags = _parse_open_options(mode);
X!   f->file_position = lseek(handle, 0L, L_INCR);
X  /* should really check first... */
END_OF_FILE
if test 435 -ne `wc -c <'fdopen.c.cdiff'`; then
    echo shar: \"'fdopen.c.cdiff'\" unpacked with wrong size!
fi
# end of 'fdopen.c.cdiff'
fi
if test -f 'fgets.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'fgets.c.cdiff'\"
else
echo shar: Extracting \"'fgets.c.cdiff'\" \(561 characters\)
sed "s/^X//" >'fgets.c.cdiff' <<'END_OF_FILE'
X*** terminator/fgets.c Fri Aug 11 18:52:10 1989
X--- new-lib/fgets.c    Thu Mar  2 22:33:20 1989
X***************
X*** 2,3 ****
X--- 2,4 ----
X  /* fgets */
X+ /* repaired somewhat by ERS -- gets didn't used to work */
X
X***************
X*** 68,71 ****
X  {
X!   fgets(buf, 99999999, stdin);
X!   return(buf);
X  }
X--- 69,80 ----
X  {
X!   char *t;
X!   if ( fgets(buf, 99999999, stdin) ) {
X!         for (t = buf; *t; t++) ;
X!         if ((t > buf) && (*(t-1) == '\n')) {
X!            --t; *t = '\0';
X!         }
X!         return(buf);
X!   }
X!   else
X!       return NULL;
X  }
END_OF_FILE
if test 561 -ne `wc -c <'fgets.c.cdiff'`; then
    echo shar: \"'fgets.c.cdiff'\" unpacked with wrong size!
fi
# end of 'fgets.c.cdiff'
fi
if test -f 'fhandle.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'fhandle.c'\"
```

```
else
echo shar: Extracting \"'fhandle.c'\" \(217 characters\)
sed "s/^X//" >'fhandle.c' <<'END_OF_FILE'
X/* file handle stat keeper, prevents excessive calls to isatty */
X
X#include "std-guts.h"
X#include "file.h"
X
Xchar    __handle_stat[N_HANDLES];   /* for isatty and fcntl */
Xchar    *__handle_to_name[N_HANDLES];    /* for fstat */
END_OF_FILE
if test 217 -ne `wc -c <'fhandle.c'`; then
    echo shar: \"'fhandle.c'\" unpacked with wrong size!
fi
# end of 'fhandle.c'
fi
if test -f 'files.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'files.c.cdiff'\"
else
echo shar: Extracting \"'files.c.cdiff'\" \(584 characters\)
sed "s/^X//" >'files.c.cdiff' <<'END_OF_FILE'
X*** terminator/files.c Fri Aug 11 18:52:13 1989
X--- new-lib/files.c    Tue Aug 15 16:59:04 1989
X***************
X*** 75,77 ****
X  /* too bad we don't have package FS here... */
X! _close_all_files()
X  {
X--- 75,79 ----
X  /* too bad we don't have package FS here... */
X! /* we don't really need to close all open files because the operating */
X! /* system does this job for us when we exit */
X! _fflush_all_files()
X  {
X***************
X*** 81,83 ****
X    if (_all_open_files[i])
X!       fclose(_all_open_files[i]);
X  }
X--- 83,85 ----
X    if (_all_open_files[i])
X!       fflush(_all_open_files[i]);
X  }
END_OF_FILE
if test 584 -ne `wc -c <'files.c.cdiff'`; then
    echo shar: \"'files.c.cdiff'\" unpacked with wrong size!
fi
# end of 'files.c.cdiff'
fi
if test -f 'fprintf.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'fprintf.c'\"
else
echo shar: Extracting \"'fprintf.c'\" \(419 characters\)
```

```
sed "s/^X//" >'fprintf.c' <<'END_OF_FILE'
X#include <stdarg.h>
X#include <stdio.h>
X
Xstatic void __fprintf_putc(c, f)
Xchar c;
XFILE * f;
X{
X  fputc(c, f);
X}
X
Xvoid fprintf(f, template, args)
XFILE * f;
Xchar * template;
Xlong args;
X{
X  _printf_guts(template, &args, __fprintf_putc, f);
X  if (isatty(fileno(f)))
X    fflush(f);
X}
X
Xvoid
Xvfprintf(FILE *f, const char *format, va_list arg)
X{
X    _printf_guts(format, arg, __fprintf_putc, f);
X    if (isatty(fileno(f)))
X        fflush(f);
X}
END_OF_FILE
if test 419 -ne `wc -c <'fprintf.c'`; then
    echo shar: \"'fprintf.c'\" unpacked with wrong size!
fi
# end of 'fprintf.c'
fi
if test -f 'fputc.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'fputc.c.cdiff'\"
else
echo shar: Extracting \"'fputc.c.cdiff'\" \(551 characters\)
sed "s/^X//" >'fputc.c.cdiff' <<'END_OF_FILE'
X*** terminator/fputc.c Fri Aug 11 18:52:16 1989
X--- new-lib/fputc.c    Wed Aug 16 00:50:36 1989
X***************
X*** 1,2 ****
X-
X  /* fputc */
X--- 1 ----
X***************
X*** 10,12 ****
X--- 9,15 ----
X  {
X+   long new_pos;
X+
X    if (!f || !f->open_p || ((f->flags & 0x03) == O_RDONLY)) return(EOF);
```

```
X+   if (f->flags & O_APPEND && !(f->flags & O_DIRTY))
X+         fseek(f, 0L, L_XTND); /* ++ sb */
X    if (f->buf_index >= BUFSIZE)
X***************
X*** 17,18 ****
X--- 20,22 ----
X    f->buf_index++;
X+   f->flags |= O_DIRTY;
X    if (f->buf_max < f->buf_index)
END_OF_FILE
if test 551 -ne `wc -c <'fputc.c.cdiff'`; then
    echo shar: \"'fputc.c.cdiff'\" unpacked with wrong size!
fi
# end of 'fputc.c.cdiff'
fi
if test -f 'freopen.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'freopen.c.cdiff'\"
else
echo shar: Extracting \"'freopen.c.cdiff'\" \(585 characters\)
sed "s/^X//" >'freopen.c.cdiff' <<'END_OF_FILE'
X*** terminator/freopen.c   Fri Aug 11 18:52:18 1989
X--- new-lib/freopen.c  Tue Aug 15 23:33:20 1989
X***************
X*** 1,3 ****
X-
X-
X  /* freopen */
X--- 1 ----
X***************
X*** 32,34 ****
X    new_flags = _parse_open_options(mode);
X!   new_handle = open(pathname, new_flags, 0);
X    if (new_handle < 0)
X--- 30,32 ----
X    new_flags = _parse_open_options(mode);
X!   new_handle = open(pathname, new_flags, 0644);
X    if (new_handle < 0)
X***************
X*** 41,42 ****
X--- 39,42 ----
X    f->last_file_error = 0;
X+   f->file_position = lseek(new_handle, 0L, L_INCR);
X+   f->eof_p = 0;
X
END_OF_FILE
if test 585 -ne `wc -c <'freopen.c.cdiff'`; then
    echo shar: \"'freopen.c.cdiff'\" unpacked with wrong size!
fi
# end of 'freopen.c.cdiff'
fi
if test -f 'ftell.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'ftell.c'\"
else
```

```
echo shar: Extracting \"'ftell.c'\" \(159 characters\)
sed "s/^X//" >'ftell.c' <<'END_OF_FILE'
X
X/* ftell */
X
X#include "std-guts.h"
X
Xint ftell(f)
Xstruct file * f;
X{
X  if (!f || !f->open_p)
X    return(-1);
X    else
X    return(f->file_position + f->buf_index);
X}
END_OF_FILE
if test 159 -ne `wc -c <'ftell.c'`; then
    echo shar: \"'ftell.c'\" unpacked with wrong size!
fi
# end of 'ftell.c'
fi
if test -f 'getpage.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'getpage.c'\"
else
echo shar: Extracting \"'getpage.c'\" \(35 characters\)
sed "s/^X//" >'getpage.c' <<'END_OF_FILE'
Xint
Xgetpagesize()
X{
X    return 256;
X}
END_OF_FILE
if test 35 -ne `wc -c <'getpage.c'`; then
    echo shar: \"'getpage.c'\" unpacked with wrong size!
fi
# end of 'getpage.c'
fi
if test -f 'getpid.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'getpid.c'\"
else
echo shar: Extracting \"'getpid.c'\" \(103 characters\)
sed "s/^X//" >'getpid.c' <<'END_OF_FILE'
X
X/* this is, of course, complete bullshit */
X
Xint getpid()
X{
X  extern long     _base;
X
X  return(_base);
X}
X
END_OF_FILE
```

```
if test 103 -ne `wc -c <'getpid.c'`; then
    echo shar: \"'getpid.c'\" unpacked with wrong size!
fi
# end of 'getpid.c'
fi
if test ! -d 'include' ; then
    echo shar: Creating directory \"'include'\"
    mkdir 'include'
fi
if test -f 'include/assert.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/assert.h'\"
else
echo shar: Extracting \"'include/assert.h'\" \(382 characters\)
sed "s/^X//" >'include/assert.h' <<'END_OF_FILE'
X#undef assert
X
X#ifdef NDEBUG
X#define assert(ignore)
X#else
X
X#define assert(expression)  \
X  do { if (!(expression)) __assert ((expression), __FILE__, __LINE__); } while
(0)
X
X#define __assert(expression, file, line)  \
X  { _eprintf ("Failed assertion " #expression     \
X          " at line %d of `" file "'.\n", line);   \
X    abort (); }
X
Xvoid _eprintf ();     /* Defined in gnulib */
X
X#endif
END_OF_FILE
if test 382 -ne `wc -c <'include/assert.h'`; then
    echo shar: \"'include/assert.h'\" unpacked with wrong size!
fi
# end of 'include/assert.h'
fi
if test -f 'include/param.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/param.h'\"
else
echo shar: Extracting \"'include/param.h'\" \(531 characters\)
sed "s/^X//" >'include/param.h' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X * $Header: param.h,v 1.1 88/01/29 17:22:30 m68k Exp $
X *
X * $Log: param.h,v $
X * Revision 1.1  88/01/29  17:22:30  m68k
X * Initial revision
X *
```

```
X */
X#define  MAXPATHLEN    259      /* 256 + 2 for a: and 1 for null */
X#define MAXHOSTNAMELEN MAXPATHLEN
X#define  NOFILE        20       /* max. number of open files/proc */
X#define  NCARGS        127      /* max number of chars in argv */
X#define  CANBSIZ       NCARGS        /* max # chars of input on one line */
END_OF_FILE
if test 531 -ne `wc -c <'include/param.h'`; then
    echo shar: \"'include/param.h'\" unpacked with wrong size!
fi
# end of 'include/param.h'
fi
if test -f 'include/sgtty.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/sgtty.h'\"
else
echo shar: Extracting \"'include/sgtty.h'\" \(395 characters\)
sed "s/^X//" >'include/sgtty.h' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X * $Header: sgtty.h,v 1.1 88/02/03 20:14:18 m68k Exp $
X *
X * $Log: sgtty.h,v $
X * Revision 1.1  88/02/03  20:14:18  m68k
X * Initial revision
X *
X */
X#ifndef _IOCTL_
X# include <ioctl.h>
X#endif  /* _IOCTL_ */
X
X#define gtty(x,y) ioctl(x,TIOCGETP,y)
X#define stty(x,y) ioctl(x,TIOCSETP,y)
END_OF_FILE
if test 395 -ne `wc -c <'include/sgtty.h'`; then
    echo shar: \"'include/sgtty.h'\" unpacked with wrong size!
fi
# end of 'include/sgtty.h'
fi
if test -f 'include/stdarg.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/stdarg.h'\"
else
echo shar: Extracting \"'include/stdarg.h'\" \(446 characters\)
sed "s/^X//" >'include/stdarg.h' <<'END_OF_FILE'
X/* this is not right, e.g. if there is a char param it takes up 4 Bytes on */
X/* the stack and this is not sizeof(char) --> these functions should better */
X/* be inline functions (er) */
X
Xtypedef char * va_list;
X#define va_start(args, last_nonvararg) args = ((char * ) (&last_nonvararg) +
sizeof(last_nonvararg))
X#define va_end(args)
```

```
X#define va_arg(args, elt_type) ((elt_type * ) (args += sizeof(elt_type)))[-1]
X#define va_dcl    va_list va_alist;
END_OF_FILE
if test 446 -ne `wc -c <'include/stdarg.h'`; then
    echo shar: \"'include/stdarg.h'\" unpacked with wrong size!
fi
# end of 'include/stdarg.h'
fi
if test ! -d 'include/sys' ; then
    echo shar: Creating directory \"'include/sys'\"
    mkdir 'include/sys'
fi
if test -f 'include/sys/dir.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/sys/dir.h'\"
else
echo shar: Extracting \"'include/sys/dir.h'\" \(17 characters\)
sed "s/^X//" >'include/sys/dir.h' <<'END_OF_FILE'
X#include <dir.h>
END_OF_FILE
if test 17 -ne `wc -c <'include/sys/dir.h'`; then
    echo shar: \"'include/sys/dir.h'\" unpacked with wrong size!
fi
# end of 'include/sys/dir.h'
fi
if test -f 'include/sys/errno.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/sys/errno.h'\"
else
echo shar: Extracting \"'include/sys/errno.h'\" \(19 characters\)
sed "s/^X//" >'include/sys/errno.h' <<'END_OF_FILE'
X#include <errno.h>
END_OF_FILE
if test 19 -ne `wc -c <'include/sys/errno.h'`; then
    echo shar: \"'include/sys/errno.h'\" unpacked with wrong size!
fi
# end of 'include/sys/errno.h'
fi
if test -f 'include/sys/file.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/sys/file.h'\"
else
echo shar: Extracting \"'include/sys/file.h'\" \(18 characters\)
sed "s/^X//" >'include/sys/file.h' <<'END_OF_FILE'
X#include <file.h>
END_OF_FILE
if test 18 -ne `wc -c <'include/sys/file.h'`; then
    echo shar: \"'include/sys/file.h'\" unpacked with wrong size!
fi
# end of 'include/sys/file.h'
fi
if test -f 'include/sys/ioctl.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/sys/ioctl.h'\"
else
echo shar: Extracting \"'include/sys/ioctl.h'\" \(19 characters\)
sed "s/^X//" >'include/sys/ioctl.h' <<'END_OF_FILE'
```

```
X#include <ioctl.h>
END_OF_FILE
if test 19 -ne `wc -c <'include/sys/ioctl.h'`; then
    echo shar: \"'include/sys/ioctl.h'\" unpacked with wrong size!
fi
# end of 'include/sys/ioctl.h'
fi
if test -f 'include/sys/param.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/sys/param.h'\"
else
echo shar: Extracting \"'include/sys/param.h'\" \(19 characters\)
sed "s/^X//" >'include/sys/param.h' <<'END_OF_FILE'
X#include <param.h>
END_OF_FILE
if test 19 -ne `wc -c <'include/sys/param.h'`; then
    echo shar: \"'include/sys/param.h'\" unpacked with wrong size!
fi
# end of 'include/sys/param.h'
fi
if test -f 'include/sys/signal.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/sys/signal.h'\"
else
echo shar: Extracting \"'include/sys/signal.h'\" \(20 characters\)
sed "s/^X//" >'include/sys/signal.h' <<'END_OF_FILE'
X#include <signal.h>
END_OF_FILE
if test 20 -ne `wc -c <'include/sys/signal.h'`; then
    echo shar: \"'include/sys/signal.h'\" unpacked with wrong size!
fi
# end of 'include/sys/signal.h'
fi
if test -f 'include/sys/stat.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/sys/stat.h'\"
else
echo shar: Extracting \"'include/sys/stat.h'\" \(18 characters\)
sed "s/^X//" >'include/sys/stat.h' <<'END_OF_FILE'
X#include <stat.h>
END_OF_FILE
if test 18 -ne `wc -c <'include/sys/stat.h'`; then
    echo shar: \"'include/sys/stat.h'\" unpacked with wrong size!
fi
# end of 'include/sys/stat.h'
fi
if test -f 'include/sys/time.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/sys/time.h'\"
else
echo shar: Extracting \"'include/sys/time.h'\" \(18 characters\)
sed "s/^X//" >'include/sys/time.h' <<'END_OF_FILE'
X#include <time.h>
END_OF_FILE
if test 18 -ne `wc -c <'include/sys/time.h'`; then
    echo shar: \"'include/sys/time.h'\" unpacked with wrong size!
fi
```

```
# end of 'include/sys/time.h'
fi
if test -f 'include/sys/times.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/sys/times.h'\"
else
echo shar: Extracting \"'include/sys/times.h'\" \(184 characters\)
sed "s/^X//" >'include/sys/times.h' <<'END_OF_FILE'
X#ifndef SYS_TIMES_H
X#define SYS_TIMES_H
X#ifndef TYPES_H
X#include <types.h>
X#endif
X
Xstruct tms {
X    time_t   tms_utime;
X    time_t   tms_stime;
X    time_t   tms_cutime;
X    time_t   tms_cstime;
X};
X#endif
END_OF_FILE
if test 184 -ne `wc -c <'include/sys/times.h'`; then
    echo shar: \"'include/sys/times.h'\" unpacked with wrong size!
fi
# end of 'include/sys/times.h'
fi
if test -f 'include/sys/types.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/sys/types.h'\"
else
echo shar: Extracting \"'include/sys/types.h'\" \(19 characters\)
sed "s/^X//" >'include/sys/types.h' <<'END_OF_FILE'
X#include <types.h>
END_OF_FILE
if test 19 -ne `wc -c <'include/sys/types.h'`; then
    echo shar: \"'include/sys/types.h'\" unpacked with wrong size!
fi
# end of 'include/sys/types.h'
fi
if test -f 'include/time.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/time.h'\"
else
echo shar: Extracting \"'include/time.h'\" \(629 characters\)
sed "s/^X//" >'include/time.h' <<'END_OF_FILE'
X#ifndef _TIME_H
X#define _TIME_H
X#include <types.h>
X
X#define CLK_TCK   200
X
Xstruct tm {
X    int  tm_sec;       /* 0..59 (on ST only the even values) */
X    int  tm_min;       /* 0..59 */
X    int  tm_hour;  /* 0..23 */
```

```
X    int  tm_mday; /* day in month 1..31 */
X    int  tm_mon;       /* month 0..11 */
X    int  tm_year; /* year - 1970 */
X    int  tm_wday; /* day in week 0..6 (0 = sunday) */
X    int  tm_yday; /* day in year 0..365 */
X    int  tm_isdst;/* daylight saving time */
X};
X
Xextern struct tm  *gmtime(const time_t * const timep);
Xextern struct tm  *localtime(const time_t * const timep);
Xextern char       *asctime(struct tm * t);
Xextern clock_t         clock(void);
X#endif /* _TIME_H */
END_OF_FILE
if test 629 -ne `wc -c <'include/time.h'`; then
    echo shar: \"'include/time.h'\" unpacked with wrong size!
fi
# end of 'include/time.h'
fi
if test -f 'include/times.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/times.h'\"
else
echo shar: Extracting \"'include/times.h'\" \(199 characters\)
sed "s/^X//" >'include/times.h' <<'END_OF_FILE'
X#ifndef SYS_TIMES_H
X#define SYS_TIMES_H
X#ifndef TYPES_H
X#include <types.h>
X#endif
X
Xstruct tms {
X    time_t   tms_utime;
X    time_t   tms_stime;
X    time_t   tms_cutime;
X    time_t   tms_cstime;
X};
X
X#define  HZ   60
X#endif
END_OF_FILE
if test 199 -ne `wc -c <'include/times.h'`; then
    echo shar: \"'include/times.h'\" unpacked with wrong size!
fi
# end of 'include/times.h'
fi
if test -f 'include/types.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/types.h'\"
else
echo shar: Extracting \"'include/types.h'\" \(596 characters\)
sed "s/^X//" >'include/types.h' <<'END_OF_FILE'
X#ifndef _TYPES_H
X/*
X *      Cross Development System for Atari ST
```

```
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X * $Header: types.h,v 1.1 88/01/29 17:22:35 m68k Exp $
X *
X * $Log: types.h,v $
X * Revision 1.1  88/01/29  17:22:35  m68k
X * Initial revision
X *
X */
Xtypedef unsigned char  u_char;
Xtypedef unsigned short u_short;
Xtypedef unsigned long  u_long;
X
Xtypedef u_long        ino_t;
Xtypedef short         dev_t;
Xtypedef long      off_t;
Xtypedef long      time_t;
Xtypedef u_long        clock_t;
Xtypedef u_short       uid_t;
Xtypedef u_short       gid_t;
Xtypedef char *        caddr_t;
X
X#define _TYPES_H
X#endif  /* types.h */
END_OF_FILE
if test 596 -ne `wc -c <'include/types.h'`; then
    echo shar: \"'include/types.h'\" unpacked with wrong size!
fi
# end of 'include/types.h'
fi
if test -f 'link.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'link.c'\"
else
echo shar: Extracting \"'link.c'\" \(70 characters\)
sed "s/^X//" >'link.c' <<'END_OF_FILE'
Xint link (const char * fromname, const char * toname)
X{
X    return(1);
X}
END_OF_FILE
if test 70 -ne `wc -c <'link.c'`; then
    echo shar: \"'link.c'\" unpacked with wrong size!
fi
# end of 'link.c'
fi
if test -f 'lseek.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'lseek.c'\"
else
echo shar: Extracting \"'lseek.c'\" \(343 characters\)
sed "s/^X//" >'lseek.c' <<'END_OF_FILE'
X/* seek to a position in a file */
X
X#include <osbind.h>
```

```
X
X#ifdef DEBUG
Xextern int stderr;
X#endif
X
Xint lseek(fd, offset, mode)
Xint fd;
Xlong offset;
Xint mode;
X{
X  int result;
X
X  if (fd == 1234) return(0);
X  result = Fseek(offset, fd, mode);
X#ifdef DEBUG
X  fprintf(stderr, "lseek(%d, %d, %d)->%d\n", fd, offset, mode);
X#endif
X  return(result);
X}
END_OF_FILE
if test 343 -ne `wc -c <'lseek.c'`; then
    echo shar: \"'lseek.c'\" unpacked with wrong size!
fi
# end of 'lseek.c'
fi
if test -f 'mkdir.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'mkdir.c'\"
else
echo shar: Extracting \"'mkdir.c'\" \(194 characters\)
sed "s/^X//" >'mkdir.c' <<'END_OF_FILE'
X#include <osbind.h>
X
Xint mkdir(dir, useless_eunuchs_ism)
Xchar * dir;
Xint useless_eunuchs_ism;
X{
X  char  name[strlen(dir)+1];
X
X  strcpy(name, dir);
X  unix_to_tos(name);
X  return(Dcreate(name));
X}
END_OF_FILE
if test 194 -ne `wc -c <'mkdir.c'`; then
    echo shar: \"'mkdir.c'\" unpacked with wrong size!
fi
# end of 'mkdir.c'
fi
if test -f 'parse-op.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'parse-op.c'\"
else
echo shar: Extracting \"'parse-op.c'\" \(567 characters\)
sed "s/^X//" >'parse-op.c' <<'END_OF_FILE'
```

```
X#include <fcntl.h>
X
Xshort _parse_open_options(options)
Xchar * options;
X{
X  int open_flags = 0;
X  int extra = 0;
X/* this version's better, but still not quite right... */
X  for ( ; *options ; options++)
X    switch (*options)
X        {
X        case 'w':
X             extra |= O_CREAT | O_TRUNC;
X             open_flags = O_WRONLY;
X             break;
X        case 'r':
X             open_flags = O_RDONLY;
X             break;
X        case 'a':
X             open_flags = O_WRONLY;
X             extra |= O_CREAT | O_APPEND;
X             break;
X        case '+':
X             open_flags = O_RDWR;
X             break;
X        case 'b':
X             extra |= FILE_BINARY_P;
X             break;
X        }
X  return(open_flags | extra);
X}
END_OF_FILE
if test 567 -ne `wc -c <'parse-op.c'`; then
    echo shar: \"'parse-op.c'\" unpacked with wrong size!
fi
# end of 'parse-op.c'
fi
if test -f 'perror.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'perror.c'\"
else
echo shar: Extracting \"'perror.c'\" \(299 characters\)
sed "s/^X//" >'perror.c' <<'END_OF_FILE'
X/* more guesswork and kludgery... */
X
X#include <errno.h>
X#include <stdio.h>
X
Xvoid perror(msg)
Xchar * msg;
X{
X  if (msg) fprintf(stderr, "%s: ", msg);
X  if ((errno > sys_nerr) || (errno < 0))
X    fprintf(stderr, "Unknown error");
```

```
X    else
X    fprintf(stderr, sys_errlist[errno]);
X  fprintf(stderr, "\n");
X}
END_OF_FILE
if test 299 -ne `wc -c <'perror.c'`; then
    echo shar: \"'perror.c'\" unpacked with wrong size!
fi
# end of 'perror.c'
fi
if test -f 'rand.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'rand.c'\"
else
echo shar: Extracting \"'rand.c'\" \(139 characters\)
sed "s/^X//" >'rand.c' <<'END_OF_FILE'
X#include <stdio.h>
X#include <osbind.h>
X
X#define RAND_MAX  0x7FFF
X
Xrand()
X{
X    return((Random() >> 3) & RAND_MAX);
X}
X
Xsrand()
X{
X    /* empty */
X}
END_OF_FILE
if test 139 -ne `wc -c <'rand.c'`; then
    echo shar: \"'rand.c'\" unpacked with wrong size!
fi
# end of 'rand.c'
fi
if test -f 'random.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'random.c'\"
else
echo shar: Extracting \"'random.c'\" \(204 characters\)
sed "s/^X//" >'random.c' <<'END_OF_FILE'
Xstatic unsigned long _seed = 1;
X
Xint random()
X    {
X    _seed = (_seed * 1103515245) + 12345;
X    return((unsigned int) ((_seed / 65536) % 32768));
X    }
X
Xvoid srandom(seed)
X    unsigned int seed;
X    {
X    _seed = seed;
X    }
```

```
X
END_OF_FILE
if test 204 -ne `wc -c <'random.c'`; then
    echo shar: \"'random.c'\" unpacked with wrong size!
fi
# end of 'random.c'
fi
if test -f 'readdir.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'readdir.c.cdiff'\"
else
echo shar: Extracting \"'readdir.c.cdiff'\" \(461 characters\)
sed "s/^X//" >'readdir.c.cdiff' <<'END_OF_FILE'
X*** terminator/readdir.c   Fri Aug 11 18:52:51 1989
X--- new-lib/readdir.c  Fri Jan 20 10:01:52 1989
X***************
X*** 25,26 ****
X--- 25,28 ----
X
X+ static ino_t    inode;
X+
X  struct direct   *
X***************
X*** 35,37 ****
X         strlen(strcpy(dp.d_name, dirp->dd_cp->_d_entry));
X!   dp.d_ino = 0;
X    dirp->dd_cp = dirp->dd_cp->_d_next;
X--- 37,39 ----
X         strlen(strcpy(dp.d_name, dirp->dd_cp->_d_entry));
X!   dp.d_ino = ++inode;
X    dirp->dd_cp = dirp->dd_cp->_d_next;
END_OF_FILE
if test 461 -ne `wc -c <'readdir.c.cdiff'`; then
    echo shar: \"'readdir.c.cdiff'\" unpacked with wrong size!
fi
# end of 'readdir.c.cdiff'
fi
if test -f 'rename.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'rename.c'\"
else
echo shar: Extracting \"'rename.c'\" \(606 characters\)
sed "s/^X//" >'rename.c' <<'END_OF_FILE'
X#include "file.h"
X#include "osbind.h"
X
X#define alloca    __builtin_alloca
X
X#ifdef DEBUG
Xextern int stderr;
X#endif
X
Xint rename(old, new)
Xchar * old, * new;
X{
```

```
X  int result;
X  char  name1[strlen(old)+1], *name2;
X
X  name2 = alloca(strlen(new)+1);
X  strcpy(name1, old);
X  strcpy(name2, new);
X  unix_to_tos(name1);
X  unix_to_tos(name2);
X#ifdef DEBUG
X  fprintf(stderr, "  rename('%s', '%s')", name1, name2);
X#endif
X  Fdelete(name2);          /* necessary? */
X  result = Frename(name1, name2);
X#ifdef DEBUG
X  fprintf(stderr, " -> %d\n");
X#endif
X  if (result > 0) result = 0;   /* what does this mean??? */
X  return (result);
X}
END_OF_FILE
if test 606 -ne `wc -c <'rename.c'`; then
    echo shar: \"'rename.c'\" unpacked with wrong size!
fi
# end of 'rename.c'
fi
if test -f 'scanf.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'scanf.c.cdiff'\"
else
echo shar: Extracting \"'scanf.c.cdiff'\" \(205 characters\)
sed "s/^X//" >'scanf.c.cdiff' <<'END_OF_FILE'
X*** terminator/scanf.c Fri Aug 11 18:52:55 1989
X--- new-lib/scanf.c   Wed Aug 16 19:06:35 1989
X***************
X*** 3,4 ****
X--- 3,7 ----
X
X+ #define TRUE    (1)
X+ #define FALSE   (0)
X+
X  extern char _numstr[];
END_OF_FILE
if test 205 -ne `wc -c <'scanf.c.cdiff'`; then
    echo shar: \"'scanf.c.cdiff'\" unpacked with wrong size!
fi
# end of 'scanf.c.cdiff'
fi
if test -f 'signame.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'signame.c'\"
else
echo shar: Extracting \"'signame.c'\" \(21 characters\)
sed "s/^X//" >'signame.c' <<'END_OF_FILE'
X#include <siglist.h>
END_OF_FILE
```

```
if test 21 -ne `wc -c <'signame.c'`; then
    echo shar: \"'signame.c'\" unpacked with wrong size!
fi
# end of 'signame.c'
fi
if test -f 'strcpy.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'strcpy.c.cdiff'\"
else
echo shar: Extracting \"'strcpy.c.cdiff'\" \(406 characters\)
sed "s/^X//" >'strcpy.c.cdiff' <<'END_OF_FILE'
X*** terminator/strcpy.c    Fri Aug 11 18:53:10 1989
X--- new-lib/strcpy.c  Mon Jan  2 16:00:20 1989
X***************
X*** 10,12 ****
X--- 10,18 ----
X   *
X+  * Revision 1.2  88/12/16  20:05:00    stefan brueck
X+  *    if (!s1 || !s2)
X+  *         return NULL;
X+  *
X   */
X+ #define NULL 0L;
X+
X  char *
X***************
X*** 17,18 ****
X--- 23,27 ----
X    char *saves1 = s1;
X+
X+   if (!s1 || !s2)             /* ..sb */
X+        return NULL;
X
END_OF_FILE
if test 406 -ne `wc -c <'strcpy.c.cdiff'`; then
    echo shar: \"'strcpy.c.cdiff'\" unpacked with wrong size!
fi
# end of 'strcpy.c.cdiff'
fi
if test -f 'strlen.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'strlen.c.cdiff'\"
else
echo shar: Extracting \"'strlen.c.cdiff'\" \(169 characters\)
sed "s/^X//" >'strlen.c.cdiff' <<'END_OF_FILE'
X*** terminator/strlen.c    Fri Aug 11 18:53:11 1989
X--- new-lib/strlen.c  Fri Mar  3 16:13:58 1989
X***************
X*** 17,18 ****
X--- 17,19 ----
X
X+   if(s)
X    while (*s++)
END_OF_FILE
if test 169 -ne `wc -c <'strlen.c.cdiff'`; then
```

```
    echo shar: \"'strlen.c.cdiff'\" unpacked with wrong size!
fi
# end of 'strlen.c.cdiff'
fi
if test -f 'times.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'times.c'\"
else
echo shar: Extracting \"'times.c'\" \(472 characters\)
sed "s/^X//" >'times.c' <<'END_OF_FILE'
X#include <sys/times.h>
X#include <time.h>
X#define HZ   60
X
Xextern clock_t    _sys_runtime; /* defined in system.c */
Xextern clock_t    _child_runtime;    /* defined in pexecv.c */
X
Xtimes(struct tms *buffer)
X{
X    if (!buffer) return;
X    buffer->tms_cutime = (time_t) (_child_runtime * HZ / CLK_TCK);
X    buffer->tms_cstime = (time_t) 0;
X    buffer->tms_stime = (time_t) (_sys_runtime * HZ / CLK_TCK);
X    buffer->tms_utime = (time_t)
X         ((clock() - _sys_runtime - _child_runtime) * HZ / CLK_TCK);
X}
END_OF_FILE
if test 472 -ne `wc -c <'times.c'`; then
    echo shar: \"'times.c'\" unpacked with wrong size!
fi
# end of 'times.c'
fi
if test -f 'uid.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'uid.c'\"
else
echo shar: Extracting \"'uid.c'\" \(35 characters\)
sed "s/^X//" >'uid.c' <<'END_OF_FILE'
Xsetuid() {}
Xgetuid() { return 0; }
END_OF_FILE
if test 35 -ne `wc -c <'uid.c'`; then
    echo shar: \"'uid.c'\" unpacked with wrong size!
fi
# end of 'uid.c'
fi
if test -f 'unlink.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'unlink.c'\"
else
echo shar: Extracting \"'unlink.c'\" \(170 characters\)
sed "s/^X//" >'unlink.c' <<'END_OF_FILE'
X#include <osbind.h>
X
Xint unlink(filename)
Xchar * filename;
```

```
X{
X  char  name[strlen(filename) +1];
X
X  strcpy(name, filename);
X  unix_to_tos(name);
X  return(Fdelete(name));
X}
END_OF_FILE
if test 170 -ne `wc -c <'unlink.c'`; then
    echo shar: \"'unlink.c'\" unpacked with wrong size!
fi
# end of 'unlink.c'
fi
if test -f 'utime.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'utime.c'\"
else
echo shar: Extracting \"'utime.c'\" \(643 characters\)
sed "s/^X//" >'utime.c' <<'END_OF_FILE'
X#include <time.h>
X#include <osbind.h>
X#include <errno.h>
X
Xlong
X_tos_time(struct tm *buf)
X{
X    long res;
X
X    /* first the time */
X    res = (buf->tm_hour << 11)|(buf->tm_min << 5)|(buf->tm_sec >> 1);
X    res <<= 16;
X    /* and then the date */
X    res |= ((buf->tm_year - 80) << 9)|((buf->tm_mon + 1) << 5)|(buf->tm_mday);
X    return(res);
X}
X
Xint
Xutime(char *name, time_t toset[2])
X{
X    struct   tm   *tm_buf;
X    long tos_time;
X    int  fd;
X
X    tm_buf = localtime(&toset[1]);   /* modification time */
X    /* convert to TOS format */
X    tos_time = _tos_time(tm_buf);
X    if((fd = Fopen(name,1)) < 0) {
X         errno = -fd;
X         return(-1);
X    }
X    Fdatime(&tos_time,fd,1);
X    Fclose(fd);
X    return(0);
X}
```

```
END_OF_FILE
if test 643 -ne `wc -c <'utime.c'`; then
    echo shar: \"'utime.c'\" unpacked with wrong size!
fi
# end of 'utime.c'
fi
if test -f 'vsprintf.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'vsprintf.c'\"
else
echo shar: Extracting \"'vsprintf.c'\" \(194 characters\)
sed "s/^X//" >'vsprintf.c' <<'END_OF_FILE'
X#include <stdarg.h>
X
Xextern void  __sprintf_pushc(char c, char **str_p);
X
Xvoid
Xvsprintf(char *s, const char *format, va_list arg)
X{
X    _printf_guts(format, arg, __sprintf_pushc, &s);
X    *s = '\0';
X}
END_OF_FILE
if test 194 -ne `wc -c <'vsprintf.c'`; then
    echo shar: \"'vsprintf.c'\" unpacked with wrong size!
fi
# end of 'vsprintf.c'
fi
echo shar: End of archive 1 \(of 7\).
cp /dev/null ark1isdone
MISSING=""
for I in 1 2 3 4 5 6 7 ; do
    if test ! -f ark${I}isdone ; then
     MISSING="${MISSING} ${I}"
    fi
done
if test "${MISSING}" = "" ; then
    echo You have unpacked all 7 archives.
    rm -f ark[1-9]isdone
else
    echo You still need to unpack the following archives:
    echo "        " ${MISSING}
fi
##  End of shell archive.
exit 0
```

```
From -7562164690557066568
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,4a163c6296370d85,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-08-19 10:53:15 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!csd4.csd.uwm.edu!cs.utexas.edu!
sun-barr!sun!grapevine!koreth%panarthea.ebay@sun.com
```

From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i073: gcclib2 -- Patches to GCC library part02/07
Keywords: shar, C
Message-ID: <34135@grapevine.uucp>
Date: 19 Aug 89 17:53:15 GMT
Sender: news@grapevine.uucp
Lines: 1421
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: uunet.UU.NET!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 2, Issue 73
Archive-name: gcclib2/part02

```
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of archive 2 (of 7)."
# Contents:  atexit.c close.c closedir.c.cdiff console.c.cdiff
#   convert.c ctime.c fclose.c fflush.c fopen.c getwd.c
#   include/ctype.h include/dir.h include/fcntl.h include/file.h
#   include/ioctl.h include/siglist.h include/sysvars.h
#   include/tchars.h isatty.c.cdiff localtim.c pause.c sfguts.c.cdiff
#   std-guts.h strstr.c tchars.c trap-14.c
# Wrapped by roeder@sbsvax on Wed Aug 16 22:02:58 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'atexit.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'atexit.c'\"
else
echo shar: Extracting \"'atexit.c'\" \(671 characters\)
sed "s/^X//" >'atexit.c' <<'END_OF_FILE'
X/* we allow atexit-registration of an arbitrary number of functions */
X/* ANSI C requires at least 32 -> we should reserve memory for this amount */
X/* but hopefully we can always get what we want */
X
Xextern struct func_list {
X    void          (*func)(void);
X    struct func_list   *next_entry;
X} *_atexit_list;  /* defined in exit.c */
X
Xint
Xatexit(void (*func)(void))
X{
X    struct func_list   *new_entry;
X    extern char        *malloc();
X
X    if(new_entry = (struct func_list *) malloc(sizeof(struct func_list))) {
X         new_entry->func = func;
```

```
X        new_entry->next_entry = _atexit_list;
X        _atexit_list = new_entry;
X        return(0);    /* successfull registration */
X    } else return(-1); /* registration failed */
X}
END_OF_FILE
if test 671 -ne `wc -c <'atexit.c'`; then
    echo shar: \"'atexit.c'\" unpacked with wrong size!
fi
# end of 'atexit.c'
fi
if test -f 'close.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'close.c'\"
else
echo shar: Extracting \"'close.c'\" \(745 characters\)
sed "s/^X//" >'close.c' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X * $Header: close.c,v 1.2 88/01/29 17:30:58 m68k Exp $
X *
X * $Log: close.c,v $
X * Revision 1.1  88/01/29  17:30:58  m68k
X * Initial revision
X *
X * 1.2  jrd
X *
X */
X#include <osbind.h>
X#include <file.h>
X
X#ifdef DEBUG
Xextern int stderr;
X#endif
X
Xint close(fd)
X    int  fd;
X{
X    extern   int  errno;
X    int      rval;
X
X    if (fd == 1234) return(0);  /* /dev/null */
X    if ((rval = Fclose(fd)) < 0) {
X        errno = -rval;
X        rval = -1;
X    }
X    if ((fd >= 0) && (fd < N_HANDLES)) {
X        __handle_stat[fd] = FH_UNKNOWN;
X        free(__handle_to_name[fd]);
X        __handle_to_name[fd] = (char *) 0;
X    }
X#ifdef DEBUG
```

```
X    fprintf(stderr, "close(%d)->%d\n", fd, rval);
X#endif
X    return rval;
X}
END_OF_FILE
if test 745 -ne `wc -c <'close.c'`; then
    echo shar: \"'close.c'\" unpacked with wrong size!
fi
# end of 'close.c'
fi
if test -f 'closedir.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'closedir.c.cdiff'\"
else
echo shar: Extracting \"'closedir.c.cdiff'\" \(627 characters\)
sed "s/^X//" >'closedir.c.cdiff' <<'END_OF_FILE'
X*** terminator/closedir.c  Fri Aug 11 18:51:54 1989
X--- new-lib/closedir.c Sat Jul 15 13:39:30 1989
X***************
X*** 22,23 ****
X--- 22,24 ----
X  static void free_dircontents();
X+ extern DIR *_current_open_directory;
X
X***************
X*** 27,29 ****
X--- 28,45 ----
X  {
X+   DIR  *dp;
X+
X    free_dircontents(dirp->dd_contents);
X+   free(dirp->dd_name);
X+   dp = _current_open_directory;
X+   if(dp == dirp) {
X+        _current_open_directory = dp->dd_next;
X+   } else {
X+        while(dp->dd_next) {
X+             if(dp->dd_next == dirp) {
X+                  dp->dd_next = dirp->dd_next;
X+                  break;
X+             }
X+             dp = dp->dd_next;
X+        }
X+   }
X    free((char *) dirp);
END_OF_FILE
if test 627 -ne `wc -c <'closedir.c.cdiff'`; then
    echo shar: \"'closedir.c.cdiff'\" unpacked with wrong size!
fi
# end of 'closedir.c.cdiff'
fi
if test -f 'console.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'console.c.cdiff'\"
else
```

```
echo shar: Extracting \"'console.c.cdiff'\" \(941 characters\)
sed "s/^X//" >'console.c.cdiff' <<'END_OF_FILE'
X*** terminator/console.c   Fri Aug 11 18:51:55 1989
X--- new-lib/console.c  Tue Aug 15 21:08:16 1989
X***************
X*** 1,2 ****
X-
X  /* console IO stuff.  Abstracted out here cause I'm having so much trouble
X--- 1 ----
X***************
X*** 10,11 ****
X--- 9,16 ----
X     Sigh.
X+
X+    There seem to be some people in the atari st community who have never
X+    heard about the st's character set. There are some chars > 0x7F which
X+    can be printed. And they will be printed.
X+    This code assumes that nowbody tries to read from stderr if stdin is
X+    redirected to AUX:.
X  */
X***************
X*** 20,23 ****
X
X!   Fread(gem_handle, 1L, &c[0]);
X!   return(c[0] & 0xFF);
X  }
X--- 25,29 ----
X
X!   return(Bconin(2));     /* isatty(gem_handle) == TRUE */
X! /*  Fread(gem_handle, 1L, &c[0]);
X!   return(c[0] & 0xFF); */
X  }
X***************
X*** 44,46 ****
X  {
X!   return(0);
X  /*
X--- 50,53 ----
X  {
X!   return(Bconstat(2));
X! /*  return(0); */
X  /*
END_OF_FILE
if test 941 -ne `wc -c <'console.c.cdiff'`; then
    echo shar: \"'console.c.cdiff'\" unpacked with wrong size!
fi
# end of 'console.c.cdiff'
fi
if test -f 'convert.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'convert.c'\"
else
echo shar: Extracting \"'convert.c'\" \(1007 characters\)
sed "s/^X//" >'convert.c' <<'END_OF_FILE'
```

```
X/* convert a unix-style filename to its TOS equivalent
X * /dev/A/dir/file.ext --> A:\DIR\FILE.EXT
X *
X * (er) : Added check for root directories (name ends after '/dev/A').
X *
X */
X
X#include <ctype.h>
X
Xchar *
Xunix_to_tos(char *fname)
X{
X    char *p;
X
X    if(fname) {
X         for(p = fname; *p; p++)
X              if(*p == '/') *p = '\\';
X              else if(islower(*p)) *p = toupper(*p);
X         if(!strncmp(fname,"\\DEV\\",5)) {
X              if(!strcmp(fname+5,"TTY")) {
X                   strcpy(fname,"CON:");
X              } else if(fname[5] && !fname[6] || fname[6] == '\\') {
X                   strcpy(fname+1,fname+5);
X                   fname[0] = fname[1];
X                   fname[1] = ':';
X              }
X         }
X    }
X    return(fname);
X}
X
X/* convert a TOS-style filename to its unix equivalent
X * A:\DIR\FILE.EXT --> a:/dir/file.ext
X * i would like to do: A:\DIR\FILE.EXT --> /dev/A/dir/file.ext, but this
X * would require to allocate room for the bigger string
X *
X */
Xchar *
Xtos_to_unix(char *fname)
X{
X    char *p;
X
X    if(fname) for(p = fname; *p; p++)
X         if(*p == '\\') *p = '/';
X         else if(isupper(*p)) *p = tolower(*p);
X    return(fname);
X}
END_OF_FILE
if test 1007 -ne `wc -c <'convert.c'`; then
    echo shar: \"'convert.c'\" unpacked with wrong size!
fi
# end of 'convert.c'
fi
```

```
if test -f 'ctime.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'ctime.c'\"
else
echo shar: Extracting \"'ctime.c'\" \(887 characters\)
sed "s/^X//" >'ctime.c' <<'END_OF_FILE'
X/*
X *  char *ctime (zeiger auf(anzahl der sekunden seit 1.1.1970 0.00 GMT))
X *  die anzahl der sekunden liefert normalerweise time()
X *
X *  rueckgabe: ein string der laenge 26 bytes im format
X *  DDD MMM dd hh:mm:ss YYYY\n\0
X *  bsp: Wed Sep 04 15:13:00 1985\n\0
X *
X *  stefan brueck 31/12/88
X */
X
X#include <time.h>
X
Xchar ctime_buf[26];        /* is it safe to use a static one? */
X
Xchar * month_name[] = {"???", "Jan", "Feb", "Mar", "Apr", "May", "Jun",
X                "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"};
Xchar * day_name[] = {"Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"};
X
Xchar * ctime(date_and_time)
Xlong * date_and_time;
X{
X    struct tm *ut;
X
X    ut = localtime(date_and_time);
X    sprintf(ctime_buf, "%s %s %02d %02d:%02d:%02d %04d\n",day_name[ut->tm_wday],
month_name[(ut->tm_mon)+1], ut->tm_mday, ut->tm_hour, ut->tm_min, (int) ut-
>tm_sec, ut->tm_year + 1900);
X
X    return((char * )ctime_buf);
X}
X
END_OF_FILE
if test 887 -ne `wc -c <'ctime.c'`; then
    echo shar: \"'ctime.c'\" unpacked with wrong size!
fi
# end of 'ctime.c'
fi
if test -f 'fclose.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'fclose.c'\"
else
echo shar: Extracting \"'fclose.c'\" \(956 characters\)
sed "s/^X//" >'fclose.c' <<'END_OF_FILE'
X/* fclose */
X/*
X * 4/1/88 changed by stefan brueck:
X */
X
```

```
X#include <errno.h>
X#include <file.h>
X#include "std-guts.h"
X
Xint fclose (f)
Xstruct file * f;
X{
X  if (f) {
X    int i;
X
X    if (f->open_p) {
X      if (((f->flags & 0x03) == O_WRONLY) ||
X      ((f->flags & 0x03) == O_RDWR))
X      {
X          /* see if buffer needs flushing */
X          int len = f->buf_max;
X          if (len > 0)
X          {
X               /*
X                * ..sb
X                * jump to the place where this buffer came from
X                */
X               if (!isatty(f->handle))
X                   lseek (f->handle, f->file_position, 0);
X               /*
X                *  now go on like before
X                */
X               write(f->handle, &f->buf, len);
X               f->buf_index = f->buf_max = 0;
X          }
X      }
X      if ((f->handle > 2) || (!isatty(f->handle)))
X          close(f->handle);
X      _remove_open_file(f);
X    }
X  }
X  if (f && (f != stdin) && (f != stdout) && (f != stderr))
X    free(f);
X
X  return(0);      /* really should check for err ... */
X}
END_OF_FILE
if test 956 -ne `wc -c <'fclose.c'`; then
    echo shar: \"'fclose.c'\" unpacked with wrong size!
fi
# end of 'fclose.c'
fi
if test -f 'fflush.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'fflush.c'\"
else
echo shar: Extracting \"'fflush.c'\" \(1047 characters\)
sed "s/^X//" >'fflush.c' <<'END_OF_FILE'
X/* fflush */
```

```
X
X#include <file.h>
X#include "std-guts.h"
X
Xextern int   _fflush_all_files();
X
Xint
Xfflush (f)
Xstruct file * f;
X{
X  if (!f) return(_fflush_all_files());
X  if (f->open_p &&
X    (((f->flags & 0x03) == O_WRONLY) || ((f->flags & 0x03) == O_RDWR)))
X    {
X    int err = 0;
X    int new_pos;
X    int len = f->buf_max;
X
X    if (len > 0)
X         {
X         if (f->flags & O_APPEND) {
X              new_pos = lseek(f->handle, 0L, L_XTND);
X              if (new_pos >= 0)
X                   f->file_position = new_pos;
X         } else lseek(f->handle, f->file_position, L_SET);
X         if (f->flags & O_DIRTY) {
X              /* the dirty flag is cleared before write to avoid */
X              /* loops if fflush gets recursively called by exit */
X              /* if ^C is pressed during this write */
X              f->flags &= ~O_DIRTY;
X              err = _write_internal(f->handle, &f->buf, len,
X                             f->flags & FILE_BINARY_P);
X              if (err < 0) {
X                   f->last_file_error = err;
X                   f->flags |= O_DIRTY;
X                   return (EOF);
X              }
X         }
X         f->last_file_error = 0;
X         }
X    f->file_position += len;
X    lseek(f->handle, f->file_position, L_SET);
X    f->buf_index = f->buf_max = 0;
X    }
X    return(0);
X}
END_OF_FILE
if test 1047 -ne `wc -c <'fflush.c'`; then
    echo shar: \"'fflush.c'\" unpacked with wrong size!
fi
# end of 'fflush.c'
fi
if test -f 'fopen.c' -a "${1}" != "-c" ; then
```

```
  echo shar: Will not clobber existing file \"'fopen.c'\"
else
echo shar: Extracting \"'fopen.c'\" \(795 characters\)
sed "s/^X//" >'fopen.c' <<'END_OF_FILE'
X/* fopen */
X
X#include <file.h>
X#include <errno.h>
X#include "std-guts.h"
X
Xstruct file * fopen (name, mode)
Xchar * name;
Xchar * mode;
X{
X  int handle;
X  struct file * f;
X  int open_flags;
X
X  open_flags = _parse_open_options(mode);
X  handle = open (name, open_flags, 0664); /* try to open it */
X  if (handle <= 0)
X    {
X      errno = -handle;
X      return(NULL);                 /* couldn't open */
X    }
X    else
X    {
X    f = (struct file * )malloc(sizeof (struct file));
X    f->handle = handle;
X    f->open_p = 1;
X    f->eof_p = 0;
X    f->last_file_error = 0;
X    f->flags = open_flags;
X    f->buf_index = 0;       /* so far... */
X    f->buf_max = 0;             /* nothing in buf yet */
X    f->file_position = 0;       /* nothing read yet */
X/* zzz should really check for file length here, for cases when we're
X   appending... */
X
X    _add_open_file(f);
X    return(f);
X    }
X
X}
END_OF_FILE
if test 795 -ne `wc -c <'fopen.c'`; then
    echo shar: \"'fopen.c'\" unpacked with wrong size!
fi
# end of 'fopen.c'
fi
if test -f 'getwd.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'getwd.c'\"
else
```

```
echo shar: Extracting \"'getwd.c'\" \(661 characters\)
sed "s/^X//" >'getwd.c' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X * $Header: getwd.c,v 1.1 88/01/29 17:29:47 m68k Exp $
X *
X * $Log: getwd.c,v $
X * Revision 1.1  88/01/29  17:29:47  m68k
X * Initial revision
X *
X * jrd 1.2
X *   initial ver was all fucked.  re-wrote.
X *
X */
X#include <osbind.h>
X#include <string.h>
X#include <param.h>
X
Xchar    *
Xgetwd(char *buf)
X{
X    int      rval;
X    int      drv;
X    char     *p;
X
X    drv = Dgetdrv();
X    if ((rval = Dgetpath(buf + 6, drv + 1)) < 0) {
X        return (char *) 0;
X    }
X    strcpy(buf,"/dev/");
X    buf[5] = drv + 'A';
X    tos_to_unix(buf+6);
X
X    return buf;
X}
X
Xchar    *
Xgetcwd(char *buf)
X{
X    return(getwd(buf));
X}
END_OF_FILE
if test 661 -ne `wc -c <'getwd.c'`; then
    echo shar: \"'getwd.c'\" unpacked with wrong size!
fi
# end of 'getwd.c'
fi
if test -f 'include/ctype.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/ctype.h'\"
else
echo shar: Extracting \"'include/ctype.h'\" \(1052 characters\)
```

```
sed "s/^X//" >'include/ctype.h' <<'END_OF_FILE'
X/*
X *  ctype.h
X *
X * Patterned after the Eunuchs one by jrd
X * 12/30/86
X *
X */
X
X                   /* upcase alphabetic */
X#define  _U   01
X                   /* lowcase alpha */
X#define  _L   02
X                   /* numeric */
X#define  _N   04
X                   /* whitespace */
X#define  _S   010
X                   /* printable */
X#define _P   020
X                   /* control */
X#define _C   040
X                   /* hex digit */
X#define _X   0100
X
Xextern  char _ctype_[];
X
X#define  isalpha(c)    ((_ctype_)[c]&(_U|_L))
X#define  isupper(c)    ((_ctype_)[c]&_U)
X#define  islower(c)    ((_ctype_)[c]&_L)
X#define  isdigit(c)    ((_ctype_)[c]&_N)
X#define  isxdigit(c)   ((_ctype_)[c]&(_N|_X))
X#define  isspace(c)    ((_ctype_)[c]&_S)
X#define ispunct(c)    ((_ctype_)[c]&_P)
X#define isalnum(c)    ((_ctype_)[c]&(_U|_L|_N))
X#define isprint(c)    ((_ctype_)[c]&(_P|_U|_L|_N))
X#define iscntrl(c)    ((_ctype_)[c]&_C)
X#define isascii(c)    ((unsigned)(c)<=0177)
X#define _toupper(c)   ((c)-'a'+'A')
X#define _tolower(c)   ((c)-'A'+'a')
X#define toascii(c)    ((c)&0177)
X
Xstatic __inline int toupper(int c) { return (islower(c) ? _toupper(c) : c); }
Xstatic __inline int tolower(int c) { return (isupper(c) ? _tolower(c) : c); }
END_OF_FILE
if test 1052 -ne `wc -c <'include/ctype.h'`; then
    echo shar: \"'include/ctype.h'\" unpacked with wrong size!
fi
# end of 'include/ctype.h'
fi
if test -f 'include/dir.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/dir.h'\"
else
echo shar: Extracting \"'include/dir.h'\" \(1362 characters\)
```

```
sed "s/^X//" >'include/dir.h' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X *  A public domain implementation of BSD directory routines for
X *  MS-DOS.  Written by Michael Rendell ({uunet,utai}michael@garfield),
X *  August 1897
X *
X * $Header: dir.h,v 1.1 88/01/29 17:22:25 m68k Exp $
X *
X * $Log: dir.h,v $
X * Revision 1.1  88/01/29  17:22:25  m68k
X * Initial revision
X *
X */
X#ifndef _DIR_H
X#define _DIR_H
X#ifndef _TYPES_H
X#include <types.h>
X#endif
X
X#define  rewinddir(dirp)    seekdir(dirp, 0L)
X
X#define  MAXNAMLEN 12
X
Xstruct direct {
X    ino_t    d_ino;             /* a bit of a farce */
X    int  d_reclen;     /* more farce */
X    int  d_namlen;     /* length of d_name */
X    char d_name[MAXNAMLEN + 1];      /* garentee null termination */
X};
X
Xstruct _dircontents {
X    char *_d_entry;
X    struct _dircontents    *_d_next;
X    u_short  _d_date;
X    u_short  _d_time;
X    char _d_attr;
X    long _d_size;
X};
X
Xtypedef struct _dirdesc {
X    int      dd_id;    /* uniquely identify each open directory */
X    long     dd_loc;   /* where we are in directory entry is this */
X    struct _dircontents    *dd_contents; /* pointer to contents of dir */
X    struct _dircontents    *dd_cp;   /* pointer to current position */
X    char     *dd_name;
X    struct _dirdesc    *dd_next;
X} DIR;
X
Xextern  DIR      *opendir();
Xextern  struct direct *readdir();
```

```
Xextern  void     seekdir();
Xextern  long     telldir();
Xextern  void     closedir();
X#endif /* _DIR_H */
END_OF_FILE
if test 1362 -ne `wc -c <'include/dir.h'`; then
    echo shar: \"'include/dir.h'\" unpacked with wrong size!
fi
# end of 'include/dir.h'
fi
if test -f 'include/fcntl.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/fcntl.h'\"
else
echo shar: Extracting \"'include/fcntl.h'\" \(988 characters\)
sed "s/^X//" >'include/fcntl.h' <<'END_OF_FILE'
X#ifndef _FCNTL_H
X#define _FCNTL_H
X
X#define  O_RDONLY 0
X#define  O_WRONLY 1
X#define  O_RDWR        2
X#define  O_NDELAY 04   /* Non-blocking I/O */
X#define  O_APPEND 010  /* append (writes guaranteed at the end) */
X#define O_SYNC         020  /* synchronous write option */
X#define O_DIRTY        0200 /* buffer was changed (needs flushing) */
X
X#define  O_CREAT       00400    /* create file (uses third open arg) */
X#define  O_TRUNC       01000    /* truncate file */
X#define  O_EXCL        02000    /* exclusive open */
X
X#define FILE_BINARY_P 040  /* file opened for binary access */
X#define FILE_FLUSH_P  0100
X
X#define  F_DUPFD       0    /* Duplicate fildes */
X#define  F_GETFD       1    /* Get fildes flags */
X#define  F_SETFD       2    /* Set fildes flags */
X#define  F_GETFL       3    /* Get file flags */
X#define  F_SETFL       4    /* Set file flags */
X#define  F_GETLK       5    /* Get file lock */
X#define  F_SETLK       6    /* Set file lock */
X#define  F_SETLKW 7    /* Set file lock and wait */
X
X#define  F_RDLCK       1
X#define  F_WRLCK       2
X#define  F_UNLCK       3
X
Xextern int fcntl (int, int, int);
X
X#endif
END_OF_FILE
if test 988 -ne `wc -c <'include/fcntl.h'`; then
    echo shar: \"'include/fcntl.h'\" unpacked with wrong size!
fi
```

```
# end of 'include/fcntl.h'
fi
if test -f 'include/file.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/file.h'\"
else
echo shar: Extracting \"'include/file.h'\" \(692 characters\)
sed "s/^X//" >'include/file.h' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X * $Header: file.h,v 1.1 88/01/29 17:22:27 m68k Exp $
X *
X * $Log: file.h,v $
X * Revision 1.1  88/01/29  17:22:27  m68k
X * Initial revision
X *
X */
X
X#ifndef FILE_H
X#define FILE_H
X/*
X *  open
X */
X
X#include <fcntl.h>
X
X/*
X * access
X */
X#define  F_OK          0
X#define  X_OK          1
X#define  W_OK          2
X#define  R_OK          4
X
X/*
X * lseek
X */
X#define  L_SET             0
X#define  L_INCR            1
X#define  L_XTND            2
X
X/* defs for handle frob */
X
X#define N_HANDLES 64
X
Xextern char __handle_stat[];
Xextern char *__handle_to_name[];
X
X#define FH_UNKNOWN    0
X#define FH_ISATTY 1
X#define FH_ISAFILE    2
X
```

```
X/* maybe more later */
X#endif
END_OF_FILE
if test 692 -ne `wc -c <'include/file.h'`; then
    echo shar: \"'include/file.h'\" unpacked with wrong size!
fi
# end of 'include/file.h'
fi
if test -f 'include/ioctl.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/ioctl.h'\"
else
echo shar: Extracting \"'include/ioctl.h'\" \(1572 characters\)
sed "s/^X//" >'include/ioctl.h' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X *  Some fields in these structures are igored, like the t_suspc/dsuspc and
X * t_startc/stopc and t_flushc.  Should add an extra field somewhere for
X * an extra erase character so both ^H and ^? would work.
X *
X * $Header: ioctl.h,v 1.1 88/02/03 20:14:16 m68k Exp $
X *
X * $Log: ioctl.h,v $
X * Revision 1.1  88/02/03  20:14:16  m68k
X * Initial revision
X *
X */
X#ifndef _IOCTL_
X# define _IOCTL_
X
X#define  TIOCGETP  0x0000        /* sgttyb */
X#define  TIOCSETP  0x0001
X#define  TIOCGETC  0x0002        /* tchars */
X#define  TIOCSETC  0x0003
X#define  TIOCGLTC  0x0004        /* ltchars */
X#define  TIOCSLTC  0x0005
X#define  TIOCSETN  0x0006        /* used in emacs, is this ok ? */
X
Xstruct tchars {
X    char t_intrc;
X    char t_quitc;
X    char t_startc;
X    char t_stopc;
X    char t_eofc;
X    char t_brkc;
X};
X
Xstruct ltchars {
X    char t_suspc;
X    char t_dsuspc;
X    char t_rprntc;
X    char t_flushc;
```

```
X    char t_werasc;
X    char t_lnextc;
X};
X
X#define  CRMOD         0x0001
X#define  CBREAK        0x0002
X#define  ECHO      0x0004
X#define  XTABS         0x0008
X#define  RAW       0x0010
X#define  LCASE         0x0000        /* does nothing (and never will!) */
X#define  TANDEM        0x0000        /* not needed */
X#define META      0x0100        /* extension: Alternate as meta key */
X
X/* Only ones that are likely to be used are here */
X#define  B300      0
X#define  B1200         1
X#define  B2400         2
X#define  B4800         3
X#define  B9600         4
X
X#define ANYP      (B300|B1200|B2400|B4800|B9600)
X#define ALLDELAY  0
X
Xstruct sgttyb {
X    char sg_ispeed;
X    char sg_ospeed;
X    char sg_erase;
X    char sg_kill;
X    short    sg_flags;
X};
X#endif  /* _IOCTL_ */
END_OF_FILE
if test 1572 -ne `wc -c <'include/ioctl.h'`; then
    echo shar: \"'include/ioctl.h'\" unpacked with wrong size!
fi
# end of 'include/ioctl.h'
fi
if test -f 'include/siglist.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/siglist.h'\"
else
echo shar: Extracting \"'include/siglist.h'\" \(706 characters\)
sed "s/^X//" >'include/siglist.h' <<'END_OF_FILE'
X/* This file was produced by running make_siglist.  Don't edit. */
X
Xchar *sys_siglist[] = {
X  "No signal",
X  "Alarm clock",
X  "Segmentation violation",
X  "Signal 3",
X  "Illegal instruction",
X  "Signal 5",
X  "Signal 6",
X  "Trace trap",
```

```
X  "Signal 8",
X  "Signal 9",
X  "Interrupt",
X  "Quit",
X  "Stopped from keyboard",
X  "Floating point exception",
X  "Hangup",
X  "Abnormal termination",
X  "Software termination",
X  "User defined signal 1",
X  "User defined signal 2"
X};
X
Xchar *signal_names[] = {
X  "ON_EXIT",
X  "SIGALRM",
X  "SIGSEGV",
X  "0",
X  "SIGILL",
X  "0",
X  "0",
X  "SIGTRAP",
X  "0",
X  "0",
X  "SIGINT",
X  "SIGQUIT",
X  "SIGTSTP",
X  "SIGFPE",
X  "SIGHUP",
X  "SIGABRT",
X  "SIGTERM",
X  "SIGUSR1",
X  "SIGUSR2"
X};
END_OF_FILE
if test 706 -ne `wc -c <'include/siglist.h'`; then
    echo shar: \"'include/siglist.h'\" unpacked with wrong size!
fi
# end of 'include/siglist.h'
fi
if test -f 'include/sysvars.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/sysvars.h'\"
else
echo shar: Extracting \"'include/sysvars.h'\" \(843 characters\)
sed "s/^X//" >'include/sysvars.h' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X *   Addresses of system variables (these are the ones "cast in stone") -
X * add them as you need them.
X *
X * $Header: sysvars.h,v 1.2 88/02/03 22:30:14 m68k Exp $
```

```
X *
X * $Log: sysvars.h,v $
X * Revision 1.2  88/02/03  22:30:14  m68k
X * Set the high bit in address <64k - this avoids a problem (in 68k chip?)
X * that causes things not to work (movl 1212:w, d0 does not work as expected).
X *
X * Revision 1.1  88/02/03  20:04:48  m68k
X * Initial revision
X *
X */
X#define  nvblq          ((short *) 0x80000454)
X/* This should be pointer to ponter to function returning int but C doesn't
X * like dealing with such things
X */
X#define _vblqueue ((char ***) 0x80000456)
X#define  _hz_200        ((long *) 0x800004bA)
X#define  conterm        ((char *) 0x80000484)
END_OF_FILE
if test 843 -ne `wc -c <'include/sysvars.h'`; then
    echo shar: \"'include/sysvars.h'\" unpacked with wrong size!
fi
# end of 'include/sysvars.h'
fi
if test -f 'include/tchars.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/tchars.h'\"
else
echo shar: Extracting \"'include/tchars.h'\" \(772 characters\)
sed "s/^X//" >'include/tchars.h' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X *  The TC defines are indices into the __tchars array.
X *
X * $Header: tchars.h,v 1.1 88/02/03 22:58:50 m68k Exp $
X *
X * $Log: tchars.h,v $
X * Revision 1.1  88/02/03  22:58:50  m68k
X * Initial revision
X *
X */
X#define  TC_ERASE  0
X#define  TC_KILL       1
X#define  TC_INTRC  2
X#define  TC_QUITC  3
X#define  TC_EOFC       4
X#define  TC_BRKC       5
X#define  TC_RPRNTC 6
X#define  TC_WERASC 7
X#define  TC_LNEXTC 8
X#define TC_RUBOUT 9
X#define TC_SUSPC  10
X#define TC_DSUSPC 11
```

```
X#define TC_FLUSHC 12
X#define  TC_UNDEF  -1
X
Xtypedef struct {
X    unsigned char *unshift;
X    unsigned char *shift;
X    unsigned char *capslock;
X} KEYTAB;
X
Xextern  char __tchars[];
Xextern  int  __ttymode;
Xextern  KEYTAB   *__keytab;
END_OF_FILE
if test 772 -ne `wc -c <'include/tchars.h'`; then
    echo shar: \"'include/tchars.h'\" unpacked with wrong size!
fi
# end of 'include/tchars.h'
fi
if test -f 'isatty.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'isatty.c.cdiff'\"
else
echo shar: Extracting \"'isatty.c.cdiff'\" \(964 characters\)
sed "s/^X//" >'isatty.c.cdiff' <<'END_OF_FILE'
X*** terminator/isatty.c    Fri Aug 11 18:52:27 1989
X--- new-lib/isatty.c  Sat Jan 14 08:24:40 1989
X***************
X*** 13,14 ****
X--- 13,15 ----
X
X+   if (handle == 1234) return(0);   /* /dev/null */
X    if (handle < 0)
X***************
X*** 17,19 ****
X    if (__handle_stat[handle] != FH_UNKNOWN)
X!       return(__handle_stat[handle] == FH_ISATTY);
X    oldloc = Fseek (0L, handle, L_INCR); /* seek zero bytes from current loc */
X--- 18,20 ----
X    if (__handle_stat[handle] != FH_UNKNOWN)
X!       return(__handle_stat[handle] & FH_ISATTY);
X    oldloc = Fseek (0L, handle, L_INCR); /* seek zero bytes from current loc */
X***************
X*** 33,37 ****
X    if (rc)
X!       __handle_stat[handle] = FH_ISATTY;
X       else
X!       __handle_stat[handle] = FH_ISAFILE;
X    return (rc);           /* return true, false, or error */
X--- 34,38 ----
X    if (rc)
X!       __handle_stat[handle] |= FH_ISATTY;
X       else
X!       __handle_stat[handle] |= FH_ISAFILE;
X    return (rc);           /* return true, false, or error */
```

```
END_OF_FILE
if test 964 -ne `wc -c <'isatty.c.cdiff'`; then
    echo shar: \"'isatty.c.cdiff'\" unpacked with wrong size!
fi
# end of 'isatty.c.cdiff'
fi
if test -f 'localtim.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'localtim.c'\"
else
echo shar: Extracting \"'localtim.c'\" \(1372 characters\)
sed "s/^X//" >'localtim.c' <<'END_OF_FILE'
X/*
X *  struct tm *localtime (zeiger auf(sec seit 1.1.1970 0.00 GMT))
X *  die anzahl der sekunden liefert normalerweise time()
X *
X *  bsp:
X *  #include <time.h>
X *  long p;
X *  struct tm *ut;
X *       time(&p);
X *       ut = localtime(&p);
X *
X *  rueckgabe: ein zeiger auf eine struktur gemaess time.h
X *
X *  stefan brueck 4/1/89
X */
X
X#include <time.h>
X
Xchar day_in_month[] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
X
Xstruct tm * localtime(const time_t *const date_and_time)
X{
X    static struct tm _tm;
X    int month, year;
X    long day, sec;
X    int days_in_year;
X
X    day  = (long) (*date_and_time / 86400L);
X    _tm.tm_wday = (int) ((day + 4) % 7);
X    sec = *date_and_time - (day * 86400L);
X    day++;
X    for ( year = 0 ; 1 ; year++)
X    {
X         if ((year + 2) % 4)
X              days_in_year = 365;
X         else
X              days_in_year = 366;
X
X         if (day - days_in_year <= 0)
X              break;
X         day -= days_in_year;
X    }
```

```
X
X    _tm.tm_year = year + 70;
X    _tm.tm_yday = (int) day;
X
X    if (!((year + 2) % 4))
X         day_in_month[2] = (char) 29;
X    else
X         day_in_month[2] = (char) 28;
X
X    for ( month = 1 ; month < 13 ; month++)
X    {
X         if (day - (long) day_in_month[month] <= 0)
X              break;
X         day -= (long) day_in_month[month];
X    }
X
X    _tm.tm_hour = (int) (sec / 3600L);
X    sec -= _tm.tm_hour * 3600L;
X    _tm.tm_min  = (int) (sec / 60L);
X    sec -= _tm.tm_min * 60L;
X
X    _tm.tm_mon  = month - 1;
X    _tm.tm_mday = day;
X    _tm.tm_sec  = sec;
X    _tm.tm_isdst = 0;
X
X    return (&_tm);
X}
END_OF_FILE
if test 1372 -ne `wc -c <'localtim.c'`; then
    echo shar: \"'localtim.c'\" unpacked with wrong size!
fi
# end of 'localtim.c'
fi
if test -f 'pause.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'pause.c'\"
else
echo shar: Extracting \"'pause.c'\" \(795 characters\)
sed "s/^X//" >'pause.c' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *
X * $Header: pause.c,v 1.0 89/01/11 00:31:30 m68k Exp $
X *
X * $Log: pause.c,v $
X *
X * 1.0       er
X *
X * Revision 1.0  89/01/11  00:31:30  m68k
X * Initial revision
X *
X */
X/* #include  <gembios.h>   ?? */
```

```
X#include <osbind.h>
X#include <ioctl.h>
X#include <signal.h>
X#include "tchars.h"
X
Xextern volatile char  _sig_done;    /* defined in signal.c */
X
X/* pause waits forever or until a signal comes in */
Xvoid
Xpause()
X{
X    char c;
X
X    _sig_done = 0;
X    while(!_sig_done) {
X         if (Cconis()) {
X              c = Crawcin() & 0xff;
X              if ((__ttymode & META) && (Kbshift(-1) & 8)) c |= 0x80;
X              if (__ttymode & RAW) continue;
X              if ((__ttymode & CRMOD) && c == '\r') c = '\n';
X              if (c == __tchars[TC_INTRC]) raise(SIGINT);
X              else if (c == __tchars[TC_QUITC]) raise(SIGQUIT);
X         }
X    }
X}
END_OF_FILE
if test 795 -ne `wc -c <'pause.c'`; then
    echo shar: \"'pause.c'\" unpacked with wrong size!
fi
# end of 'pause.c'
fi
if test -f 'sfguts.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'sfguts.c.cdiff'\"
else
echo shar: Extracting \"'sfguts.c.cdiff'\" \(685 characters\)
sed "s/^X//" >'sfguts.c.cdiff' <<'END_OF_FILE'
X*** terminator/sfguts.c    Fri Aug 11 18:52:57 1989
X--- new-lib/sfguts.c  Wed Aug 16 19:07:02 1989
X***************
X*** 1,2 ****
X!
X  /* this version borrowed from Dlibs, and bug-fixed a bit by jrd */
X--- 1,2 ----
X! /* er: added %n format to get more ANSI compatibility */
X  /* this version borrowed from Dlibs, and bug-fixed a bit by jrd */
X***************
X*** 6,7 ****
X--- 6,10 ----
X
X+ #define TRUE    (1)
X+ #define FALSE   (0)
X+
X  /* #define skip()   do{c=(*get)(ip); if (c<1) goto done;}while(isspace(c))*/
```

```
X***************
X*** 242,243 ****
X--- 245,253 ----
X                           }
X+                      break;
X+
X+                 case 'n':
X+                      if (store) {
X+                           p = *args;
X+                           *((int *) p) = cnt;
X+                      }
X                       break;
END_OF_FILE
if test 685 -ne `wc -c <'sfguts.c.cdiff'`; then
    echo shar: \"'sfguts.c.cdiff'\" unpacked with wrong size!
fi
# end of 'sfguts.c.cdiff'
fi
if test -f 'std-guts.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'std-guts.h'\"
else
echo shar: Extracting \"'std-guts.h'\" \(1045 characters\)
sed "s/^X//" >'std-guts.h' <<'END_OF_FILE'
X
X/* internal structure defs for FILE structs and the code that hacks on them */
X
X/* buffer size */
X
X#define BUFSIZE 1024
X
Xstruct file
X    {
X    unsigned char      open_p;       /* 1-> this one's open */
X    unsigned char eof_p;        /* 1-> at eof */
X    unsigned char flags;        /* read and write mode bits */
X    unsigned char slush;
X    unsigned short     handle;       /* the file_handle GEMDOS gave us */
X    short         buf_index;    /* idx into buf; signed cause it can be
X                           -1 when somebody ungetcs something */
X    short         buf_max;  /* end of valid data in buf */
X    unsigned long file_position;     /* nbytes into file of buf */
X    int      last_file_error; /* last error encountered */
X    short         unchar_slush; /* in case we unchar at begin buf */
X    char     buf[BUFSIZE]; /* and the buffer */
X    };
X
X#define file_output_p(f)   \
X    (((f->flags & 0x03) == O_WRONLY) || ((f->flags & 0x03) == O_RDWR))
X
X#define file_input_p(f)    \
X    (((f->flags & 0x03) == O_RDONLY) || ((f->flags & 0x03) == O_RDWR))
X
X#define EOF -1
```

```
X#define NULL 0
X
Xextern struct file * stdin;
Xextern struct file * stdout;
Xextern struct file * stderr;
END_OF_FILE
if test 1045 -ne `wc -c <'std-guts.h'`; then
    echo shar: \"'std-guts.h'\" unpacked with wrong size!
fi
# end of 'std-guts.h'
fi
if test -f 'strstr.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'strstr.c'\"
else
echo shar: Extracting \"'strstr.c'\" \(1127 characters\)
sed "s/^X//" >'strstr.c' <<'END_OF_FILE'
X/*
X *      strstr() - sucht in dem string s2 den string s1
X *      rueckgabe ist entweder ein zeiger auf das erste
X *      vorkommen vom ersten zeichen von s1 in s2, falls
X *      s1 in s2 vorhanden ist, oder NULL, falls s1 in
X *      s2 nicht vorhanden ist oder einer der beiden strings
X *      ein zeiger auf '\0' ist.
X *
X *      stefan brueck 11/88
X *      diese source ist frei von jedem copyright.
X */
X
X#define NULL (char *) 0L
X
Xchar * strstr(s1,s2)
Xchar * s1;
Xchar * s2;
X{
X    register char *scan1;
X    register char *scan2;
X
X    if (!(s1 == NULL || s2 == NULL))
X    {
X        if (*s2 && *s1)
X            for (scan2 = s2 ; *scan2 ; scan2 = ++s2)
X            {
X                for (scan1 = s1; *scan1 ; scan1++)
X                {
X                    if (*scan1 != *scan2 || !*scan2)
X                        break;
X                    scan2++;
X                }
X
X            /* der zu suchende string liegt komplett in
X             * dem zu testenden string drin -> gefundene
X             * position zurueckgeben
X             */
```

```
X                  if (!*scan1)
X                       return (s2);
X
X              /* wenn nicht, dann weitersuchen beim naechsten
X                  * buchstaben des zu testendem strings, bis entweder
X               * was gefunden wurde, oder das ende vom zu durchsuchenden
X               * string erreicht wurde.
X               */
X              }
X    }
X    return (NULL);
X}
X
END_OF_FILE
if test 1127 -ne `wc -c <'strstr.c'`; then
    echo shar: \"'strstr.c'\" unpacked with wrong size!
fi
# end of 'strstr.c'
fi
if test -f 'tchars.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'tchars.c'\"
else
echo shar: Extracting \"'tchars.c'\" \(858 characters\)
sed "s/^X//" >'tchars.c' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X *  These to keep track of the current special characters and tty mode.
X *
X * $Header: tchars.c,v 1.1 88/02/03 22:58:46 m68k Exp $
X *
X * $Log: tchars.c,v $
X * Revision 1.1  88/02/03  22:58:46  m68k
X * Initial revision
X *
X */
X#include <ioctl.h>
X#include "tchars.h"
X
Xint __ttymode = CRMOD|ECHO|XTABS;
Xchar    __tchars[13] = {
X             010,      /* TC_ERASE - ^H */
X             025,      /* TC_KILL - ^U */
X             003,      /* TC_INTRC - ^C */
X             034,      /* TC_QUITC - ^\ */
X             004,      /* TC_EOFC - ^D */
X             TC_UNDEF, /* TC_BRKC - undefined */
X             022,      /* TC_RPRNTC - ^R */
X             027,      /* TC_WREASC - ^W */
X             026,      /* TC_LNEXTC - ^V */
X             0177,          /* TC_RUBOUT - RUB */
X             032,      /* TC_SUSP - ^Z */
```

```
X             031,      /* TC_DSUSP - ^Y */
X             TC_UNDEF  /* TC_FLUSHC - undefined */
X         };
XKEYTAB  *__keytab;
END_OF_FILE
if test 858 -ne `wc -c <'tchars.c'`; then
    echo shar: \"'tchars.c'\" unpacked with wrong size!
fi
# end of 'tchars.c'
fi
if test -f 'trap-14.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'trap-14.c'\"
else
echo shar: Extracting \"'trap-14.c'\" \(1472 characters\)
sed "s/^X//" >'trap-14.c' <<'END_OF_FILE'
X
X/* xbios trap */
X
Xtrap_14_wwll(w1, w2, l1, l2)
Xlong w1, w2, l1, l2;
X{
X  asm("moveml #0x3FFC,sp@-");
X  asm("movel a6@(20),sp@-");
X  asm("movel a6@(16),sp@-");
X  asm("movel a6@(12),d0");
X  asm("movew d0,sp@-");
X  asm("movel a6@(8),d0");
X  asm("movew d0,sp@-");
X  asm("bset #0, __in_trap");
X  asm("trap #14");
X  asm("bclr #0, __in_trap");
X  asm("addl #12,sp");
X  asm("moveml sp@+,#0x3FFC");
X}
X
Xtrap_14_wllw(w1, l1, l2, w2)
Xlong w1, l1, l2, w2;
X{
X  asm("moveml #0x3FFC,sp@-");
X  asm("movel a6@(20),d0");
X  asm("movew d0,sp@-");
X  asm("movel a6@(16),sp@-");
X  asm("movel a6@(12),sp@-");
X  asm("movel a6@(8),d0");
X  asm("movew d0,sp@-");
X  asm("bset #0, __in_trap");
X  asm("trap #14");
X  asm("bclr #0, __in_trap");
X  asm("addl #12,sp");
X  asm("moveml sp@+,#0x3FFC");
X}
X
Xtrap_14_wlll(w1, l1, l2, l3)
```

```
Xlong w1, l1, l2, l3;
X{
X  asm("moveml #0x3FFC,sp@-");
X  asm("movel a6@(20),sp@-");
X  asm("movel a6@(16),sp@-");
X  asm("movel a6@(12),sp@-");
X  asm("movel a6@(8),d0");
X  asm("movew d0,sp@-");
X  asm("bset #0, __in_trap");
X  asm("trap #14");
X  asm("bclr #0, __in_trap");
X  asm("addl #14,sp");
X  asm("moveml sp@+,#0x3FFC");
X}
X
Xtrap_14_www(w1, w2, w3)
Xlong w1, w2, w3;
X{
X  asm("moveml #0x3FFC,sp@-");
X  asm("movel a6@(16),d0");
X  asm("movew d0,sp@-");
X  asm("movel a6@(12),d0");
X  asm("movew d0,sp@-");
X  asm("movel a6@(8),d0");
X  asm("movew d0,sp@-");
X  asm("bset #0, __in_trap");
X  asm("trap #14");
X  asm("bclr #0, __in_trap");
X  asm("addl #6,sp");
X  asm("moveml sp@+,#0x3FFC");
X}
X
END_OF_FILE
if test 1472 -ne `wc -c <'trap-14.c'`; then
    echo shar: \"'trap-14.c'\" unpacked with wrong size!
fi
# end of 'trap-14.c'
fi
echo shar: End of archive 2 \(of 7\).
cp /dev/null ark2isdone
MISSING=""
for I in 1 2 3 4 5 6 7 ; do
    if test ! -f ark${I}isdone ; then
     MISSING="${MISSING} ${I}"
    fi
done
if test "${MISSING}" = "" ; then
    echo You have unpacked all 7 archives.
    rm -f ark[1-9]isdone
else
    echo You still need to unpack the following archives:
    echo "        " ${MISSING}
fi
```

```
##  End of shell archive.
exit 0
```

```
From -5838339067684477989
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,55c99fa65fd538ec,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-08-19 10:53:28 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!csd4.csd.uwm.edu!cs.utexas.edu!
sun-barr!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i074: gcclib2 -- Patches to GCC library  part03/07
Keywords: shar, C
Message-ID: <34136@grapevine.uucp>
Date: 19 Aug 89 17:53:28 GMT
Sender: news@grapevine.uucp
Lines: 1307
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: uunet.UU.NET!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 2, Issue 74
Archive-name: gcclib2/part03
```

```
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of archive 3 (of 7)."
# Contents:  alrm_nas.s include/errno.h include/signal.h include/stat.h
#   include/stdio.h include/stdlib.h ioctl.c.cdiff open.c
#   opendir.c.cdiff st-error.c std-init.c time.c trap-1.c trap-13.c
#   write.c.cdiff
# Wrapped by roeder@sbsvax on Wed Aug 16 22:03:00 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'alrm_nas.s' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'alrm_nas.s'\"
else
echo shar: Extracting \"'alrm_nas.s'\" \(1840 characters\)
sed "s/^X//" >'alrm_nas.s' <<'END_OF_FILE'
X|
X|       Cross Development System for Atari ST
X|      Copyright (c) 1988, Memorial University of Newfoundland
X|
X|   At this point we take over from dos - we know (think?) that the stack
X| looks like this now:
X|   0    long pc
X|   4    long arg (pointer to d7 below)
```

```
X|       (stuff - unknown size - depends on compiler - could be size 0)
X|       long pc
X|   *4   long d7
X|   *4+0x4    long a0
X|   *4+0x8    long d0
X|   *4+0xc    long d1
X|        .
X|        .
X|   *4+0x24   long d7
X|   *4+0x28   long a0
X|        .
X|        .
X|   *4+0x40   long a6
X|   *4+0x44   word sr   status register from exception
X|   *4+0x46   long pc   pc from exception
X|       stuff
X|  The stratagy at this point consists of saving the exception sr/pc and puting
X| out own address in place of the pc.  When the system does it's rte we get
X| called and we call raise (which can call some routine which can do a
X| longjmp).  If raise returns then we restore the sr/pc and do a rte.
X|
X| $Header: alrm_nasty.s,v 1.1 88/02/03 22:37:55 m68k Exp $
X|
X| $Log: alrm_nasty.s,v $
X| Revision 1.1  88/02/03  22:37:55  m68k
X| Initial revision
X|
X
X.globl  __in_trap          | extern
X.globl  _raise                 | extern
X.globl  __alrm_nasty
X.globl  __received_signal
X
X.comm   alrm_sr, 2             | static
X.comm   alrm_pc, 4             | static
X.comm   __received_signal, 4
X
X__alrm_nasty:
X    movel    sp@(0x4), a0
X    movew    a0@(0x44), alrm_sr | save sr/pc at static locations, it's
X    movel    a0@(0x46), alrm_pc | safe now because we are blocking sigs
X    movew    #0x2300, a0@(0x44) | fake sr (with super bit on)
X    movel    #alrm_fake, a0@(0x46)   | our fake return routine
X    rts
X
Xalrm_fake:
X    movel    alrm_pc, sp@-      | push original sr/pc on stack so we
X    movew    alrm_sr, sp@-      | can handle other signals during raise
X    moveml   d0-d2/a0-a2, sp@-  | save registers
X    movel    __received_signal, sp@- | raise(SIG..)
X    clrl __received_signal  | unblock other signals
X    jsr  _raise             |
X    addql    #0x4, sp       | pop signum
```

```
X    moveml   sp@+, d0-d2/a0-a2  | pop registers
X    rte
END_OF_FILE
if test 1840 -ne `wc -c <'alrm_nas.s'`; then
    echo shar: \"'alrm_nas.s'\" unpacked with wrong size!
fi
# end of 'alrm_nas.s'
fi
if test -f 'include/errno.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/errno.h'\"
else
echo shar: Extracting \"'include/errno.h'\" \(2191 characters\)
sed "s/^X//" >'include/errno.h' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X * $Header: errno.h,v 1.2 88/02/03 20:07:07 m68k Exp $
X *
X * $Log: errno.h,v $
X * Revision 1.2  88/02/03  20:07:07  m68k
X * Added ENOTTY for ioctl()
X *
X * Revision 1.1  88/01/29  17:18:36  m68k
X * Initial revision
X *
X */
X#define  ENOERR          0       /* no error */
X#define  EERROR          1       /* generic error */
X#define  EDRNRDY         2       /* drive not ready */
X#define  EUKCMD          3       /* unknown command */
X#define  ECRC        4       /* crc error */
X#define  EBADREQ         5       /* bad request */
X#define  ESEEK           6       /* seek error */
X#define  EUKMEDIA    7       /* unknown media */
X#define  ESECTOR         8       /* sector not found */
X#define  EPAPER          9       /* no paper */
X#define  EWRITE         10       /* write fault */
X#define  EREAD          11       /* read fault */
X#define  EGENERIC   12       /* general mishap */
X#define  EROFS          13       /* write protect */
X#define  ECHMEDIA   14       /* media change */
X#define  EUKDEV         15       /* unknown device */
X#define  EBADSEC        16       /* bad sectors */
X#define  EIDISK         17       /* insert disk */
X                       /* (gap) */
X#define  EINVAL         32       /* invalid function number */
X#define  ENOENT         33       /* file not found */
X#define  EPATH          34       /* path not found */
X#define  ENOTDIR        34       /* path not found */
X#define  EMFILE         35       /* too many open files */
X#define  EACCESS        36       /* access denied */
X#define  EACCES         36       /* access denied */
```

```
X#define  EBADF          37      /* invalid handle */
X#define  ENOMEM         39      /* insufficient memory */
X#define  EFAULT         40      /* invalid memory block request */
X#define  ENXIO          46      /* invalid drive */
X#define  EXDEV          48      /* cross device rename */
X#define  ENMFILES  49       /* no more files (fnext) */
X#define EDOM       62
X#define ERANGE          63
X#define  EBADARG        64      /* range error/context unknown */
X#define  EINTERNAL 65       /* internal error */
X#define  ENOEXEC        66      /* invalid program load format */
X#define  ESBLOCK        67      /* set block failed/growth restraints */
X                       /* (gap) */
X#define  EEXIST         80      /* file exists (open) */
X#define  ENAMETOOLONG   81      /* file exists (open) */
X#define  ENOTTY         82      /* not a tty (ioctl) */
X#define EAGAIN        EDRNRDY       /* try again later */
X
Xextern  int  errno;
Xextern  int  sys_nerr;
Xextern  char *   sys_errlist[];
END_OF_FILE
if test 2191 -ne `wc -c <'include/errno.h'`; then
    echo shar: \"'include/errno.h'\" unpacked with wrong size!
fi
# end of 'include/errno.h'
fi
if test -f 'include/signal.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/signal.h'\"
else
echo shar: Extracting \"'include/signal.h'\" \(2052 characters\)
sed "s/^X//" >'include/signal.h' <<'END_OF_FILE'
X#ifndef _SIGNAL_H
X#define _SIGNAL_H
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X *  Beware the SIGINT/SIGQUIT - right now they only work when you are read()ing
X *
X * $Header: signal.h,v 1.1 88/02/03 20:06:23 m68k Exp $
X *
X * cleaned up for ANSI C (er) [13.8.1989]:
X * added typedefs, SIG_ERR and a few signal values
X *
X * $Log: signal.h,v $
X * Revision 1.1  88/02/03  20:06:23  m68k
X * Initial revision
X *
X */
X#ifdef __GNUC__
X#pragma once
X#endif
```

```
X
Xtypedef long sig_atomic_t; /* the integral type of an object that can  */
X                 /* be accessed as an atomic entity, even in */
X                 /* the presence of asynchronous interrupts. */
Xtypedef int  sig_handler_t(int);    /* type for signal handlers */
X
X#define SIG_DFL       ((sig_handler_t *)  0)
X#define SIG_ERR       ((sig_handler_t *) -1)
X#define SIG_IGN       ((sig_handler_t *)  1)
X
X#define NSIG     19
X
X#define SIGNULL       0        /* The null signal */
X#define SIGALRM       1        /* If SIGALRM changes, change the value
X                       * in lib/libc/atari/alrm_nasty.s too!
X                       */
X/* this block are just trap numbers.  I made up most of these names... */
X#define SIGBUS        2
X#define SIGODD        3
X#define SIGILL        4
X#define SIGZDIV       5
X#define SIGCHK        6
X#define SIGTRAP       7
X#define SIGPRIV       8
X#define SIGTRACE  9
X
X
X#define SIGINT        10       /* ^C (struct tchars t_intc) */
X#define SIGQUIT       11       /* ^\ (struct tchars t_quitc) */
X#define SIGTSTP       12       /* ^Z (struct tchars t_suspc) */
X#define SIGFPE        13       /* floating point exception */
X#define SIGHUP        14       /* hangup */
X#define SIGABRT       15       /* abnormal termination */
X#define SIGTERM       16       /* termination request */
X#define SIGUSR1       17       /* user defined signal #1 */
X#define SIGUSR2       18       /* user defined signal #2 */
X
X/* ANSI C requires the following */
X#define SIGSEGV       SIGBUS        /* invalid access to storage */
X
X/* some values that Eunuchs programs seem to want */
X#define SIGIOT    SIGNULL
X#define SIGEMT    SIGNULL
X#define SIGSYS    SIGNULL
X#define SIGSTOP   SIGTSTP
X
Xsig_handler_t     *signal(int sig, sig_handler_t *func);
Xint      raise(int sig);
X#endif  /* _SIGNAL_H */
END_OF_FILE
if test 2052 -ne `wc -c <'include/signal.h'`; then
    echo shar: \"'include/signal.h'\" unpacked with wrong size!
fi
```

```
# end of 'include/signal.h'
fi
if test -f 'include/stat.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/stat.h'\"
else
echo shar: Extracting \"'include/stat.h'\" \(1664 characters\)
sed "s/^X//" >'include/stat.h' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X *  Fake stat the best we can.  The only field that isn't filled in half-way
X * sensibly in st_ino.  Also put dta struct here for lack of a better place.
X *
X * $Header: stat.h,v 1.1 88/01/29 17:22:33 m68k Exp $
X *
X * $Log: stat.h,v $
X * Revision 1.1  88/01/29  17:22:33  m68k
X * Initial revision
X *
X */
X#ifndef _STAT_H
X#define _STAT_H
X
X#ifndef _TYPES_H
X#include <types.h>
X#endif /* _TYPES_H */
X
Xstruct stat {
X    u_short  st_mode;
X    ino_t    st_ino;
X    dev_t    st_dev;
X    short    st_rdev;
X    short    st_nlink;
X    short    st_uid;
X    short    st_gid;
X    off_t    st_size;
X    long st_blksize;
X    long st_blocks;
X    time_t   st_mtime;
X    time_t   st_atime;
X    time_t   st_ctime;
X    short    st_attr;
X};
X
X#define  S_IFMT   0170000       /* type of file */
X#define      S_IFDIR  0040000  /* directory */
X#define      S_IFCHR  0020000  /* character special */
X#define      S_IFBLK  0060000  /* block special */
X#define      S_IFREG  0100000  /* regular */
X#define      S_IFIFO  0010000  /* fifo */
X#define  S_ISUID  04000         /* set user id on execution */
X#define  S_ISGID  02000         /* set group id on execution */
```

```
X#define S_ISVTX  01000         /* save swapped text even after use */
X#define S_IREAD  00400         /* read permission, owner */
X#define S_IWRITE 00200         /* write permission, owner */
X#define S_IEXEC  00100         /* execute/search permission, owner */
X
Xstruct _dta {
X    char     dta_buf[21];
X    char     dta_attribute;
X    unsigned short    dta_time;
X    unsigned short    dta_date;
X    long     dta_size;
X    char     dta_name[14];
X};
X
X#define FA_RDONLY     0x01
X#define FA_HIDDEN     0x02
X#define FA_SYSTEM     0x04
X#define FA_LABEL      0x08
X#define FA_DIR             0x10
X#define FA_CHANGED         0x20
X#endif /* _STAT_H */
END_OF_FILE
if test 1664 -ne `wc -c <'include/stat.h'`; then
    echo shar: \"'include/stat.h'\" unpacked with wrong size!
fi
# end of 'include/stat.h'
fi
if test -f 'include/stdio.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/stdio.h'\"
else
echo shar: Extracting \"'include/stdio.h'\" \(1990 characters\)
sed "s/^X//" >'include/stdio.h' <<'END_OF_FILE'
X#ifndef STDIO_H
X#define STDIO_H
X/*
X * stdio.h
X *
X * this version kludged up by jrd
X */
X
X/* files are really pointers to something, but that isn't visible to standard
X   code */
X#define  FILE long
X#define  EOF  -1
X
X#ifndef NULL
X#define NULL (void *)0
X#endif
X
X/* internal structure defs for FILE structs and the code that hacks on them */
X
X#define BUFSIZE 1024
X
```

```
Xstruct file
X    {
X    unsigned char      open_p;       /* 1-> this one's open */
X    unsigned char eof_p;        /* 1-> at eof */
X    unsigned char flags;        /* read and write mode bits */
X    unsigned char slush;
X    unsigned short     handle;       /* the file_handle GEMDOS gave us */
X    short         buf_index;    /* idx into buf; signed cause it can be
X                          -1 when somebody ungetcs something */
X    short         buf_max;  /* end of valid data in buf */
X    unsigned long file_position;     /* nbytes into file of buf */
X    int       last_file_error; /* last error encountered */
X    short         unchar_slush; /* in case we unchar at begin buf */
X    char      buf[BUFSIZE]; /* and the buffer */
X    };
X
X#define file_output_p(f)   \
X    (((f->mode & 0x03) == O_WRONLY) || ((f->mode & 0x03) == O_RDWR))
X
X#define file_input_p(f)    \
X    (((f->mode & 0x03) == O_RDONLY) || ((f->mode & 0x03) == O_RDWR))
X
X#undef BUFSIZE
X
X/* standard files */
Xextern struct file _std_in_out_err[3];
X#define stdin     (FILE *) &_std_in_out_err[0]
X#define stdout    (FILE *) &_std_in_out_err[1]
X#define stderr    (FILE *) &_std_in_out_err[2]
X
X/* and code to hack them */
Xextern FILE * fopen(), * freopen(), * fdopen();
X
Xextern int fileno(FILE *fp);
Xextern int fgetc();
Xextern char * fgets();
Xextern char * gets();
X
X/* more eunuchs compatibility brain damage... */
X#define putc fputc
X#define  getc fgetc
X#define putchar(c)     fputc(c, stdout)
X#define getchar() fgetc(stdin)
X
X/* this is so unbelievably brain-dead I can't think of an epithet
X   that accurately describes it.  No, it has no use at all, other
X   than to support eunuchs code that doesn't believe in modularity
X   or functional boundaries.  That is, most of it.... */
X#define BUFSIZ    512
X#endif /* STDIO_H */
END_OF_FILE
if test 1990 -ne `wc -c <'include/stdio.h'`; then
    echo shar: \"'include/stdio.h'\" unpacked with wrong size!
```

```
fi
# end of 'include/stdio.h'
fi
if test -f 'include/stdlib.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/stdlib.h'\"
else
echo shar: Extracting \"'include/stdlib.h'\" \(1705 characters\)
sed "s/^X//" >'include/stdlib.h' <<'END_OF_FILE'
X#ifndef _STDLIB_H
X#define _STDLIB_H
X
Xtypedef unsigned long  size_t;
Xtypedef char     wchar_t;
Xtypedef struct {
X        int  quot;
X        int  rem;
X    } div_t;
Xtypedef struct {
X        long quot;
X        long rem;
X    } ldiv_t;
X
X#define NULL      (void *) 0
X#define EXIT_FAILURE  1
X#define EXIT_SUCCESS  0
X#define RAND_MAX  0x7FFF
X#define  MB_CUR_MAX    1
X
Xextern double     atof(const char *nptr);
Xextern int   atoi(const char *nptr);
Xextern long int   atol(const char *nptr);
Xextern double     strtod(const char *nptr, char **endptr);
Xextern long int   strtol(const char *nptr, char **endptr, int base);
Xextern unsigned long int strtoul(const char *nptr, char **endptr, int base);
Xextern int   rand(void);
Xextern void  srand(unsigned int seed);
Xextern void  *calloc(size_t nmemb, size_t size);
Xextern void  free(void *ptr);
Xextern void  *malloc(size_t size);
Xextern void  *realloc(void *ptr, size_t size);
Xextern void  abort(void);
Xextern int   atexit(void (*func)(void));
Xextern void  exit(int status);
Xextern char  *getenv(const char *name);
Xextern int   system(const char *string);
X/* bsearch */
X/* qsort */
Xextern int   abs(int j);
X/* div */
Xextern long int   labs(long int j);
X/* ldiv */
X/* mblen */
X/* mbtowc */
```

```
X/* wctomb */
X/* mbstowcs */
X/* wcstombs */
Xextern void  *memcpy(void *s1, const void *s2, size_t n);
Xextern void  *memmove(void *s1, const void *s2, size_t n);
Xextern char  *strcpy(char *s1, const char *s2);
Xextern char  *strncpy(char *s1, const char *s2, size_t n);
Xextern char  *strcat(char *s1, const char *s2);
Xextern char  *strncat(char *s1, const char *s2, size_t n);
X/* ... */
Xextern size_t     strlen(const char *s);
X
X/* this was the standard ansi header - and now the gnu cc version of this */
X#ifdef __GNUC__
X#pragma once
X
X#define abs  __builtin_abs
X#endif
X#endif
END_OF_FILE
if test 1705 -ne `wc -c <'include/stdlib.h'`; then
    echo shar: \"'include/stdlib.h'\" unpacked with wrong size!
fi
# end of 'include/stdlib.h'
fi
if test -f 'ioctl.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'ioctl.c.cdiff'\"
else
echo shar: Extracting \"'ioctl.c.cdiff'\" \(1934 characters\)
sed "s/^X//" >'ioctl.c.cdiff' <<'END_OF_FILE'
X*** terminator/ioctl.c Fri Aug 11 18:52:27 1989
X--- new-lib/ioctl.c   Fri Jul 14 20:09:48 1989
X***************
X*** 14,15 ****
X--- 14,16 ----
X   */
X+ #include   <fcntl.h>
X  #include   <ioctl.h>
X***************
X*** 17,19 ****
X--- 18,34 ----
X  #include   "tchars.h"
X+ #include   <osbind.h>
X+ #include   <sysvars.h>
X
X+ static int
X+ meta_on()
X+ {
X+   *conterm |= 8;
X+ }
X+
X+ static int
X+ meta_off()
```

```
X+ {
X+   *conterm &= ~8;
X+ }
X+
X  int
X***************
X*** 27,28 ****
X--- 42,44 ----
X    struct ltchars     *ltc;
X+   long ssp;
X
X***************
X*** 41,43 ****
X
X!   case TIOCSETP:
X         sg = (struct sgttyb *) arg;
X--- 57,68 ----
X
X!   case TIOCSETP:     /* discard input */
X!        {
X!             int  old_delay_mode;
X!             char ch;
X!
X!             old_delay_mode = fcntl(0, F_SETFL, O_NDELAY);
X!             while (read(0, &ch, 1) == 1);
X!             fcntl(0, F_SETFL, old_delay_mode);
X!        }
X!   case TIOCSETN:
X         sg = (struct sgttyb *) arg;
X***************
X*** 45,46 ****
X--- 70,78 ----
X         __tchars[TC_KILL] = sg->sg_kill;
X+        if (sg->sg_flags & META)
X+             __keytab = (KEYTAB *) Keytbl(-1,-1,-1);
X+        if (sg->sg_flags & META) {
X+             Supexec(meta_on);
X+        } else {
X+             Supexec(meta_off);
X+        }
X         __ttymode = sg->sg_flags;
X***************
X*** 68,73 ****
X         ltc = (struct ltchars *) arg;
X!        ltc->t_suspc = TC_UNDEF;
X!        ltc->t_dsuspc = TC_UNDEF;
X         ltc->t_rprntc = __tchars[TC_RPRNTC];
X!        ltc->t_flushc = TC_UNDEF;
X         ltc->t_werasc = __tchars[TC_WERASC];
X--- 100,105 ----
X         ltc = (struct ltchars *) arg;
X!        ltc->t_suspc = __tchars[TC_SUSPC];
X!        ltc->t_dsuspc = __tchars[TC_DSUSPC];
```

```
X        ltc->t_rprntc = __tchars[TC_RPRNTC];
X!       ltc->t_flushc = __tchars[TC_FLUSHC];
X        ltc->t_werasc = __tchars[TC_WERASC];
X***************
X*** 78,80 ****
X--- 110,115 ----
X        ltc = (struct ltchars *) arg;
X+       __tchars[TC_SUSPC] = ltc->t_suspc;
X+       __tchars[TC_DSUSPC] = ltc->t_dsuspc;
X        __tchars[TC_RPRNTC] = ltc->t_rprntc;
X+       __tchars[TC_FLUSHC] = ltc->t_flushc;
X        __tchars[TC_WERASC] = ltc->t_werasc;
END_OF_FILE
if test 1934 -ne `wc -c <'ioctl.c.cdiff'`; then
    echo shar: \"'ioctl.c.cdiff'\" unpacked with wrong size!
fi
# end of 'ioctl.c.cdiff'
fi
if test -f 'open.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'open.c'\"
else
echo shar: Extracting \"'open.c'\" \(2057 characters\)
sed "s/^X//" >'open.c' <<'END_OF_FILE'
X/*
X * 	Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X * er: added O_NDELAY, /dev/null, fstat()-support
X *
X *  Tries to emulate unix open3.  The mode argument is ignored, and O_NDELAY
X * doesn't do anything.  Also O_APPEND does not garentee that things will
X * be written to the end - all we can do is seek to the end to start with.
X *
X * $Header: open.c,v 1.2 88/01/29 17:31:36 m68k Exp $
X *
X * $Log: open.c,v $
X *
X * Modified by jrd for native gcc 1.2
X *
X * Revision 1.2  88/02/24  17:31:36  m68k
X *
X */
X#include <types.h>
X#include <file.h>
X#include <stat.h>
X#include <errno.h>
X#include <osbind.h>
X
X#ifdef DEBUG
Xextern int stderr;
X#endif
X
Xint open(p, flags, mode)
```

```
Xchar    * p;
Xshort   flags;
Xu_short mode;
X{
X  char       path[strlen(p)+1];
X  int        fd;
X  int        exists;
X  extern char    *malloc();
X  u_short    create_mode = 0;
X  int   len;
X
X  if(!strcmp(p,"/dev/null")) return(1234);
X  len = strlen(p) + 1;
X  strcpy(path,p);
X  unix_to_tos(path);
X  switch (flags & 0x3) {
X  case O_RDONLY:
X    fd = Fopen(path, O_RDONLY);
X    break;
X  case O_WRONLY:
X  case O_RDWR:
X    fd = 0;
X    exists = Fattrib(path, 0, 0) >= 0;
X    if (flags & O_CREAT) {
X      if ((flags & O_EXCL) && exists)
X    fd = - EEXIST;
X    } else if (!exists) fd = - ENOENT;
X    if (!fd) {
X      if ((flags & O_TRUNC) || !exists) {
X    if (!(mode & S_IREAD))
X      create_mode |= FA_HIDDEN;
X    if (!(mode & S_IWRITE))
X      create_mode |= FA_RDONLY;
X    if ((fd = Fcreate(path, create_mode)) >= 0
X        && (flags & 0x3) == O_RDWR) {
X      (void) Fclose(fd);
X      fd = Fopen(path, O_RDWR);
X    }
X      } else if ((fd = Fopen(path, flags & 0x3)) >= 0
X          && (flags & O_APPEND))
X    (void) Fseek(0L, fd, L_XTND);
X    }
X    break;
X
X  default:
X    fd = - EINVAL;
X  }
X  if (fd < 0) {
X    errno = -fd;
X    fd = -1;
X  } else if (fd < N_HANDLES) {
X    if(flags & O_NDELAY)
X      __handle_stat[fd] |= O_NDELAY;
```

```
X    if(__handle_to_name[fd] = malloc(len))
X      strcpy(__handle_to_name[fd],p);
X  }
X#ifdef DEBUG
X  fprintf(stderr, "open('%s', %X)->%d\n", path, flags, fd);
X#endif
X  return fd;
X}
END_OF_FILE
if test 2057 -ne `wc -c <'open.c'`; then
    echo shar: \"'open.c'\" unpacked with wrong size!
fi
# end of 'open.c'
fi
if test -f 'opendir.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'opendir.c.cdiff'\"
else
echo shar: Extracting \"'opendir.c.cdiff'\" \(2772 characters\)
sed "s/^X//" >'opendir.c.cdiff' <<'END_OF_FILE'
X*** terminator/opendir.c   Fri Aug 11 18:52:45 1989
X--- new-lib/opendir.c  Tue Aug 15 17:40:08 1989
X***************
X*** 84,91 ****
X
X  DIR   *
X! opendir(name)
X!   char *name;
X  {
X    DIR           *dirp;
X-   char          c;
X    char          *s;
X--- 84,94 ----
X
X+ extern DIR *_current_open_directory; /* in stat.c */
X+ static int dirnumber = 0;
X+
X  DIR   *
X! opendir(n)
X!   char *n;
X  {
X+   char          *name;
X    DIR           *dirp;
X    char          *s;
X***************
X*** 93,95 ****
X--- 96,101 ----
X    char          nbuf[MAXPATHLEN + 1];
X+   int           len;
X
X+   name = malloc(strlen(n)+1);
X+   strcpy(name,n);
X  #ifdef MSDOS
X***************
```

```
X*** 106,107 ****
X--- 112,115 ----
X    {
X+       struct   stat     statb;
X+
X        /* This bit of nastyness is due to the fact that Fattrib()
X***************
X*** 110,111 ****
X--- 118,120 ----
X         */
X+       unix_to_tos(name);
X        if (index(name, '*') || index(name, '?'))
X***************
X*** 112,114 ****
X--- 121,128 ----
X             return (DIR *) NULL;
X+       if (stat(name, &statb) < 0
X+            || (statb.st_mode & S_IFMT) != S_IFDIR)
X+                return (DIR *) NULL;
X        setdta();
X+ #ifdef old_code
X+       /* this old code could never deal with root directories */
X        if (getdirent(name) == (char *) NULL ||
X***************
X*** 116,117 ****
X--- 130,132 ----
X                 return (DIR *) NULL;
X+ #endif
X    }
X***************
X*** 120,125 ****
X        return (DIR *) NULL;
X!   if (*name && (c = name[strlen(name) - 1]) != '\\' && c != '/')
X!       (void) strcat(strcpy(nbuf, name), "\\*.*");
X!   else
X!       (void) strcat(strcpy(nbuf, name), "*.*");
X    dirp->dd_loc = 0;
X--- 135,140 ----
X        return (DIR *) NULL;
X!   len = strlen(name) - 1;
X!   if (len >= 0 && name[len] == '\\')
X!       name[len] = '\0';
X!   (void) strcat(strcpy(nbuf, name), "\\*.*");
X    dirp->dd_loc = 0;
X***************
X*** 126,127 ****
X--- 141,155 ----
X    dirp->dd_contents = dirp->dd_cp = (struct _dircontents *) NULL;
X+   /* the following code helps optimization in stat() and fstat() */
X+   if(*name == '.' && !name[1]) {   /* directory is '.' */
X+       /* don't care about '..' or other strange variants */
X+       char namebuf[MAXPATHLEN + 1];
X+       extern char   *realloc();
```

```
X+
X+       Dgetpath(namebuf, 0);
X+       name = realloc(name, strlen(namebuf)+1);
X+       strcpy(name,namebuf);
X+   }
X+   dirp->dd_name = name;
X+   dirp->dd_next = _current_open_directory;
X+   _current_open_directory = dirp;
X    if ((s = getdirent(nbuf)) == (char *) NULL)
X***************
X*** 141,143 ****
X--- 169,176 ----
X             dirp->dd_contents = dirp->dd_cp = dp;
X+       tos_to_unix(s);
X        (void) strcpy(dp->_d_entry, s);
X+       dp->_d_attr = dtabuf.dta_attribute;
X+       dp->_d_date = dtabuf.dta_date;
X+       dp->_d_time = dtabuf.dta_time;
X+       dp->_d_size = dtabuf.dta_size;
X        dp->_d_next = (struct _dircontents *) NULL;
X***************
X*** 145,146 ****
X--- 178,180 ----
X    dirp->dd_cp = dirp->dd_contents;
X+   dirp->dd_id = ++dirnumber;
X
END_OF_FILE
if test 2772 -ne `wc -c <'opendir.c.cdiff'`; then
    echo shar: \"'opendir.c.cdiff'\" unpacked with wrong size!
fi
# end of 'opendir.c.cdiff'
fi
if test -f 'st-error.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'st-error.c'\"
else
echo shar: Extracting \"'st-error.c'\" \(1745 characters\)
sed "s/^X//" >'st-error.c' <<'END_OF_FILE'
X
X/* does this replace some Eunuchs thing?  Who knows, in this kludge of a
X * language.  In any event, input-scrub.c wants it; nobody else seems to use
X * it.
X */
X
X/* apparently Eunuchs has a similar thing, but their error numbers
X   go up from 0.  We'll hack things a bit so as to allow negative offsets
X   from sys_errlist to get the right strings here */
X
X#include <errno.h>
X
Xint sys_nerr = 67;              /* max known error number */
Xint errno = 0;                  /* current error num */
X
X/* this, of course, must follow the above immediately.  We rely
```

```
X   on GCC optimizing this into a reference to the duplicate string, above */
Xchar * sys_errlist[] =
X    {
X    "No error",
X    "Generic bogosity",         /* -1 */
X    "Drive not ready",
X    "Unknown command",
X    "CRC error",
X    "Bad request",
X    "Seek error",
X    "Unknown media",
X    "Sector not found",
X    "Paper out",
X    "Write fault",
X    "Read fault",
X    "Random bogons",
X    "Write protected",
X    "Media change",
X    "Unknown device",
X    "Bad sectors",
X    "Insert disk",
X    "", "", "", "", "", "",     /* -23 .. -18 */
X    "", "", "", "", "", "", "", "",  /* -31 .. -24 */
X    "Invalid function number",
X    "File not found",
X    "Path not found",
X    "Too many open files",
X    "Access denied",
X    "Invalid file handle",
X    "",                /* -38 */
X    "Not enough memory",
X    "Memory fault",
X    "", "", "", "", "",
X    "Invalid drive",
X    "",                /* -47 */
X    "Cross device rename",
X    "No more files",
X    "", "", "", "", "", "",     /* -55 .. -50 */
X    "", "", "", "", "", "",
X    "Math function argument error",
X    "Result out of range",  /* -63 .. -56 */
X    "Bad arg",              /* -64 */
X    "Internal error",
X    "Not an executable",
X    "Error setting block size", /* -67 */
X    "System error",             /* min-error minus 1 */
X    };
X
Xchar * sys_error(num)
Xint num;
X{
X  if ((num > sys_nerr) || (num < 0))
X    num = 0;
```

```
X  return(sys_errlist[num]);
X}
END_OF_FILE
if test 1745 -ne `wc -c <'st-error.c'`; then
    echo shar: \"'st-error.c'\" unpacked with wrong size!
fi
# end of 'st-error.c'
fi
if test -f 'std-init.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'std-init.c'\"
else
echo shar: Extracting \"'std-init.c'\" \(1695 characters\)
sed "s/^X//" >'std-init.c' <<'END_OF_FILE'
X/* this is a sort of standard init frobule for general purpose
X   c programs.  You can replace it if you want something that
X   doesn't drag in half the known universe at link time.
X*/
X
X#include <stdio.h>
X#include <file.h>
X#include <time.h>
X
X#undef stdin
X#undef stdout
X#undef stderr
X
X/* Some programs (like lex) think that they can use 'FILE *f = stdin' to */
X/* initialize f. If we collect these three file structures in an array - */
X/* we would need them anyway - these programs will not loose anymore.    */
Xstruct file _std_in_out_err[3];
Xstruct file * stdin = NULL;     /* these variables are for older programs */
Xstruct file * stdout = NULL;
Xstruct file * stderr = NULL;
X
Xstatic struct file * _fdopen(f, handle, mode)
Xstruct file * f;
Xint handle;
Xchar * mode;
X{
X  f->handle = handle;
X
X/* I guess we have to trust that the mode string passed in matches the
X   one the handle was opened with ?!?  What a crock... */
X  f->flags = _parse_open_options(mode);
X  f->file_position = 0;
X/* should really check first... */
X  f->eof_p = 0;
X  f->buf_index = 0;
X  f->buf_max = 0;
X  f->open_p = 1;
X  f->last_file_error = 0;
X}
X
```

```
Xextern clock_t    _start_time;  /* defined in clock.c */
X
Xint _main (n_args, args, env)
Xint n_args;
Xchar ** args;
Xchar * env;       /* is this right? */
X{
X  int return_code;
X
X  _start_time = clock();
X  stdin = &_std_in_out_err[0];
X  stdout = &_std_in_out_err[1];
X  stderr = &_std_in_out_err[2];
X  _fdopen(stdin, 0, "r");
X  _fdopen(stdout, 1, "a");
X  _fdopen(stderr, 2, "w+");
X
X  return_code = main(n_args, args, env);
X
X  fflush(stdin);       /* just fflush, it's closed by the OS */
X  fflush(stdout);
X  fflush(stderr);
X
X  return(return_code);     /* some programs want to return a status */
X}
END_OF_FILE
if test 1695 -ne `wc -c <'std-init.c'`; then
    echo shar: \"'std-init.c'\" unpacked with wrong size!
fi
# end of 'std-init.c'
fi
if test -f 'time.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'time.c'\"
else
echo shar: Extracting \"'time.c'\" \(1617 characters\)
sed "s/^X//" >'time.c' <<'END_OF_FILE'
X
X/* julian date */
X/*
X * time() - liefert die anzahl der sekunden seit 1.1.1970 0:00:00 GMT
X *
X * stefan brueck 12/88
X * - korrektur 1980 -> 1970
X * - schaltjahre werden beruecksichtigt
X */
X
X#include <osbind.h>
X
X/* days in months = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}; */
X
Xshort month_days[] = { 0,
X             31,
X             31 + 28,
```

```
X             31 + 28 + 31,
X             31 + 28 + 31 + 30,
X             31 + 28 + 31 + 30 + 31,
X             31 + 28 + 31 + 30 + 31 + 30,
X             31 + 28 + 31 + 30 + 31 + 30 + 31,
X             31 + 28 + 31 + 30 + 31 + 30 + 31 + 31,
X             31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30,
X             31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31,
X             31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31 + 30
X             };
X
Xlong _time(place, tostime)
Xlong * place;
Xlong tostime;
X{
X  int today, now;
X  int year, month, day, hour, min, sec;
X  int leap_years;
X  long result;
X
X  today = (tostime >> 16) & 0xFFFF;
X  now   = tostime & 0xFFFF;
X  year  = ((today >> 9) & 0x7F);    /* since 1980!! */
X  year  += 10;                  /* since 1970     */
X  month = (today >> 5) & 0x0F;
X  if ((month < 1) || (month > 12))
X    month = 1;
X  day   = today & 0x1F;
X  hour  = (now >> 11) & 0x1F;
X  min   = (now >> 5) & 0x3F;
X  sec   = (now & 0x01F) << 1;
X
X  result =   sec +
X         (min * 60) +
X         (hour * 3600) +
X         ((day-1 + month_days[month-1]) * 86400) +
X         year * 31536000;
X
X    /* schaltjahre hinzuaddieren */
X    leap_years = (year + 1) >> 2;        /* since 1970 */
X    if (!((year+2) & 3) && month > 2)    /* im schaltjahr */
X         leap_years++;
X    result += leap_years * 86400;
X
X  if (place)
X    *place = result;
X  return(result);
X}
X
Xlong time(place)
Xlong *place;
X{
X  return _time(place,Gettime());    /* XBIOS = hardwareclock/ikbd */
```

```
X}
END_OF_FILE
if test 1617 -ne `wc -c <'time.c'`; then
    echo shar: \"'time.c'\" unpacked with wrong size!
fi
# end of 'time.c'
fi
if test -f 'trap-1.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'trap-1.c'\"
else
echo shar: Extracting \"'trap-1.c'\" \(1609 characters\)
sed "s/^X//" >'trap-1.c' <<'END_OF_FILE'
Xextern char  _in_trap;
X
Xint trap_1_wwlll(args)
Xlong args;
X{
X/* arg[0] is at a6@(8) */
X
X  asm("movel a6@(24),sp@-");         /* copy l3 */
X  asm("movel a6@(20),sp@-");         /* copy l2 */
X  asm("movel a6@(16),sp@-");         /* copy l1 */
X  asm("movel a6@(12),d0");      /* copy w2 */
X  asm("movew d0,sp@-");
X  asm("movel a6@(8),d0");       /* copy w1 */
X  asm("movew d0,sp@-");
X  asm("bset #0, __in_trap");
X  asm("trap #1");          /* do the trap */
X  asm("bclr #0, __in_trap");
X  asm("addl #12,sp");           /* pop junk off stack */
X}
X
Xint trap_1_wlww(args)
Xlong args;
X{
X  asm("movel a6@(20),d0");      /* copy w3 */
X  asm("movew d0,sp@-");
X  asm("movel a6@(16),d0");      /* copy w2 */
X  asm("movew d0,sp@-");
X  asm("movel a6@(12),sp@-");         /* copy l1 */
X  asm("movel a6@(8),d0");       /* copy w1 */
X  asm("movew d0,sp@-");
X  asm("bset #0, __in_trap");
X  asm("trap #1");          /* do the trap */
X  asm("bclr #0, __in_trap");
X  asm("addl #10,sp");           /* pop junk off stack */
X}
X
Xint trap_1_www(args)
Xlong args;
X{
X  asm("movel a6@(16),d0");      /* copy w3 */
X  asm("movew d0,sp@-");
```

```
X  asm("movel a6@(12),d0");      /* copy w2 */
X  asm("movew d0,sp@-");
X  asm("movel a6@(8),d0");       /* copy w1 */
X  asm("movew d0,sp@-");
X  asm("bset #0, __in_trap");
X  asm("trap #1");          /* do the trap */
X  asm("bclr #0, __in_trap");
X  asm("addl #6,sp");            /* pop junk off stack */
X}
X
Xint trap_1_wl(args)
Xlong args;
X{
X  asm("movel a6@(12),d0");      /* copy w2 */
X  asm("movel d0,sp@-");
X  asm("movel a6@(8),d0");       /* copy w1 */
X  asm("movew d0,sp@-");
X  asm("bset #0, __in_trap");
X  asm("trap #1");          /* do the trap */
X  asm("bclr #0, __in_trap");
X  asm("addl #6,sp");            /* pop junk off stack */
X}
END_OF_FILE
if test 1609 -ne `wc -c <'trap-1.c'`; then
    echo shar: \"'trap-1.c'\" unpacked with wrong size!
fi
# end of 'trap-1.c'
fi
if test -f 'trap-13.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'trap-13.c'\"
else
echo shar: Extracting \"'trap-13.c'\" \(1792 characters\)
sed "s/^X//" >'trap-13.c' <<'END_OF_FILE'
X
Xint trap_13_www(args)
Xlong args;
X{
X/* arg[0] is at a6@(8) */
X/* push w, w, w.  more later? */
X
X  asm("moveml #0x3FFC,sp@-");       /* save all regs... */
X  asm("movel a6@(16),d0");      /* copy w1 */
X  asm("movew d0,sp@-");
X  asm("movel a6@(12),d0");      /* copy w2 */
X  asm("movew d0,sp@-");
X  asm("movel a6@(8),d0");       /* copy w1 */
X  asm("movew d0,sp@-");
X  asm("bset #0, __in_trap");
X  asm("trap #13");              /* do the trap */
X  asm("bclr #0, __in_trap");
X  asm("addl #6,sp");            /* pop junk off stack */
X  asm("moveml sp@+,#0x3FFC");       /* restore regs */
X}
```

```
X
Xint trap_13_wwl(args)
Xlong args;
X{
X/* arg[0] is at a6@(8) */
X/* push w, w, l.  more later? */
X
X  asm("moveml #0x3FFC,sp@-");       /* save all regs... */
X  asm("movel a6@(16),sp@-");        /* copy l1 */
X  asm("movel a6@(12),d0");      /* copy w2 */
X  asm("movew d0,sp@-");
X  asm("movel a6@(8),d0");       /* copy w1 */
X  asm("movew d0,sp@-");
X  asm("bset #0, __in_trap");
X  asm("trap #13");              /* do the trap */
X  asm("bclr #0, __in_trap");
X  asm("addl #8,sp");            /* pop junk off stack */
X  asm("moveml sp@+,#0x3FFC");       /* restore regs */
X}
X
X
Xint trap_13_wwlwww(args)
Xlong args;
X{
X/* arg[0] is at a6@(8) */
X/* push w, w, l.  more later? */
X
X  asm("moveml #0x3FFC,sp@-");       /* save all regs... */
X  asm("movel a6@(28),d0");      /* copy w5 */
X  asm("movew d0,sp@-");
X  asm("movel a6@(24),d0");      /* copy w4 */
X  asm("movew d0,sp@-");
X  asm("movel a6@(20),d0");      /* copy w3 */
X  asm("movew d0,sp@-");
X  asm("movel a6@(16),sp@-");        /* copy l1 */
X  asm("movel a6@(12),d0");      /* copy w2 */
X  asm("movew d0,sp@-");
X  asm("movel a6@(8),d0");       /* copy w1 */
X  asm("movew d0,sp@-");
X  asm("bset #0, __in_trap");
X  asm("trap #13");              /* do the trap */
X  asm("bclr #0, __in_trap");
X  asm("addl #14,sp");           /* pop junk off stack */
X  asm("moveml sp@+,#0x3FFC");       /* restore regs */
X}
X
END_OF_FILE
if test 1792 -ne `wc -c <'trap-13.c'`; then
    echo shar: \"'trap-13.c'\" unpacked with wrong size!
fi
# end of 'trap-13.c'
fi
if test -f 'write.c.cdiff' -a "${1}" != "-c" ; then
```

```
  echo shar: Will not clobber existing file \"'write.c.cdiff'\"
else
echo shar: Extracting \"'write.c.cdiff'\" \(1631 characters\)
sed "s/^X//" >'write.c.cdiff' <<'END_OF_FILE'
X*** terminator/write.c Fri Aug 11 18:53:23 1989
X--- new-lib/write.c    Wed Aug 16 19:11:15 1989
X***************
X*** 8,9 ****
X--- 8,14 ----
X   *
X+  * 1.4 er
X+  * Added /dev/null special case
X+  * Moved __col_pos from read.c to write.c because this code is always needed by
X+  * fflush in exit.c
X+  *
X   * 1.3 jrd
X***************
X*** 17,19 ****
X   */
X- /* #include     <gembios.h>   ?? */
X  #include <osbind.h>
X--- 22,23 ----
X***************
X*** 27,32 ****
X
X! int write(fd, buf, nbytes)
X! int fd;
X! char *buf;
X! unsigned int nbytes;
X  {
X--- 31,36 ----
X
X! int    __col_pos = 0;
X!
X! int
X! write(int fd, char *buf, unsigned int nbytes)
X  {
X***************
X*** 36,39 ****
X
X- extern int  __col_pos;
X-
X  int _write_internal(fd, buf, nbytes, binary_p)
X--- 40,41 ----
X***************
X*** 45,47 ****
X--- 47,52 ----
X    char *s;
X+   int retval;
X
X+   if (fd == 1234) return(nbytes); /* /dev/null */
X+
X  #ifdef DEBUG
```

```
X***************
X*** 49,50 ****
X--- 54,56 ----
X    fprintf(stderr, "write_internal(%d, %X, %d, %d)\n", fd, buf, nbytes,
binary_p);
X+   return Fwrite(fd, nbytes, buf);
X  #endif
X***************
X*** 62,63 ****
X--- 68,71 ----
X    }
X+
X+   retval = nbytes;
X    for (s = buf ; nbytes > 0 ; s++, nbytes-- )
X***************
X*** 82,84 ****
X            else
X!       if (*s >= ' ' && *s < 0177)
X             __col_pos++;
X--- 90,94 ----
X            else
X!       if (*s >= ' ')             /* the ST *HAS* printable chars */
X!                                   /* above 0177 (ever heard about */
X!                                   /* non american users ???) */
X             __col_pos++;
X***************
X*** 89,91 ****
X        }
X!   return nbytes;
X  }
X--- 99,101 ----
X        }
X!   return retval;
X  }
END_OF_FILE
if test 1631 -ne `wc -c <'write.c.cdiff'`; then
    echo shar: \"'write.c.cdiff'\" unpacked with wrong size!
fi
# end of 'write.c.cdiff'
fi
echo shar: End of archive 3 \(of 7\).
cp /dev/null ark3isdone
MISSING=""
for I in 1 2 3 4 5 6 7 ; do
    if test ! -f ark${I}isdone ; then
     MISSING="${MISSING} ${I}"
    fi
done
if test "${MISSING}" = "" ; then
    echo You have unpacked all 7 archives.
    rm -f ark[1-9]isdone
else
    echo You still need to unpack the following archives:
```

```
    echo "        " ${MISSING}
fi
##  End of shell archive.
exit 0
```

```
From 2275129143885811906
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,21c50a65393d978c,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-08-19 10:53:37 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!csd4.csd.uwm.edu!cs.utexas.edu!
sun-barr!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i075: gcclib2 -- Patches to GCC library  part04/07
Keywords: shar, C
Message-ID: <34137@grapevine.uucp>
Date: 19 Aug 89 17:53:37 GMT
Sender: news@grapevine.uucp
Lines: 1074
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: uunet.UU.NET!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 2, Issue 75
Archive-name: gcclib2/part04
```

```
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of archive 4 (of 7)."
# Contents:  memory.c.cdiff pexecv.c signal.c stat.c system.c timer.c
# Wrapped by roeder@sbsvax on Wed Aug 16 22:03:02 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'memory.c.cdiff' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'memory.c.cdiff'\"
else
echo shar: Extracting \"'memory.c.cdiff'\" \(4440 characters\)
sed "s/^X//" >'memory.c.cdiff' <<'END_OF_FILE'
X*** terminator/memory.c    Fri Aug 11 18:52:40 1989
X--- new-lib/memory.c  Thu Aug  3 22:37:44 1989
X***************
X*** 1 ****
X--- 1,14 ----
X+ /* Added malloc_init and warnfunction for emacs support
X+  * This will only work if you set _stksize = -1. Otherwise we have no
X+  * correct lim_data and data_bytes_unused
X+  * [[ i have a solution for this, but i included this stuff for emacs only ]]
```

```
X+  * er (Edgar Roeder)
X+  */
X+ /* Changed 89/02/06, AD (Atze Dijkstra )
X+    System request made dependant of _heapbase, init in crt0.s.
X+    If != NULL, mem will not be requested via Malloc,
X+    but heap and stack are contiguous and consume almost
X+    all the free mem.
X+    Corrected 23-Jun-1989 by Piet van Oostrum
X+  */
X
X***************
X*** 3,4 ****
X--- 16,25 ----
X
X+ extern char *        _heapbase ;   /*AD, PvO*/
X+ extern char *        _stksize ;    /*AD, PvO*/
X+
X+ static long     malloc_bytes_in_free_list = 0;
X+
X+ long  malloc_sbrk_used = 0;
X+ long  malloc_sbrk_unused = 1024000;    /* only dummy initializer */
X+
X  /* minimum chunk to ask OS for */
X***************
X*** 6,9 ****
X
X- #include <osbind.h>
X-
X  #define NULL 0
X--- 27,28 ----
X***************
X*** 20,21 ****
X--- 39,91 ----
X
X+ /* Number of bytes of writable memory we can expect to be able to get */
X+ static long lim_data = 0;
X+
X+ /* Level number of warnings already issued.
X+   0 -- no warnings issued.
X+   1 -- 75% warning already issued.
X+   2 -- 85% warning already issued.
X+ */
X+ static int warnlevel;
X+
X+ /* Function to call to issue a warning;
X+    0 means don't issue them.  */
X+ static void (*warnfunction) ();
X+
X+ /* Cause reinitialization based on job parameters;
X+   also declare where the end of pure storage is. */
X+ void
X+ malloc_init (start, warnfun)  /* ignore start - it's for emacs to get happy */
X+      char *start;
```

```
X+      void (*warnfun) ();
X+ {
X+   malloc_sbrk_unused = lim_data = 0;
X+   warnlevel = 0;
X+   if (warnfun != (void (*)) -1)
X+     warnfunction = warnfun;
X+ }
X+
X+ static char * HeapAlloc( sz )          /*AD, PvO*/
X+   unsigned long sz ;              /*AD*/
X+ {                          /*AD*/
X+
X+   char slush [64];                /*PvO*/
X+   register char* sp ;                  /*AD, PvO*/
X+   int available;
X+                            /*AD*/
X+   sp = slush;                          /*AD, PvO*/
X+   available = (int) (sp - _heapbase + sz);
X+   if (!lim_data) lim_data = available;
X+   if ( available < 0 )                 /*AD, PvO, er*/
X+   {                              /*AD*/
X+     return( NULL ) ;                   /*AD*/
X+   }                              /*AD*/
X+   malloc_sbrk_unused = available + malloc_bytes_in_free_list;
X+   malloc_bytes_in_free_list += sz;
X+   sp = _heapbase ;                /*AD*/
X+   _heapbase += sz ;               /*AD*/
X+   _stksize -= sz ;                /*AD*/
X+                            /*AD*/
X+   return( sp ) ;                  /*AD*/
X+ }                          /*AD*/
X+
X  char * malloc(n)
X***************
X*** 62,65 ****
X  */
X! /*    q = (struct mem_chunk * )trap_1_wlww(0x48, sz);
     */
X!   q = (struct mem_chunk * )Malloc(sz);
X  /*
X--- 132,137 ----
X  */
X!   if ( _heapbase )                    /*AD*/
X!        q = (struct mem_chunk * )HeapAlloc( sz ) ;     /*AD*/
X!   else                               /*AD*/
X!        q = (struct mem_chunk * )trap_1_wlww(0x48, sz);
X  /*
X***************
X*** 68,69 ****
X--- 140,162 ----
X  */
X+   if(warnfunction)
X+   switch (warnlevel) {
```

```
X+       case 0:
X+       if (malloc_sbrk_used > (lim_data / 4) * 3) {
X+            warnlevel++;
X+            (*warnfunction) ("Warning: past 75% of memory limit\n");
X+       }
X+       break;
X+       case 1:
X+       if (malloc_sbrk_used > (lim_data / 20) * 17) {
X+            warnlevel++;
X+            (*warnfunction) ("Warning: past 85% of memory limit\n");
X+       }
X+       break;
X+       case 2:
X+       if (malloc_sbrk_used > (lim_data / 20) * 19) {
X+            warnlevel++;
X+       (*warnfunction) ("Warning: past 95% of memory limit\n");
X+       }
X+       break;
X+   }
X    if (((long)q) <= 0)         /* can't alloc any more? */
X***************
X*** 94,95 ****
X--- 187,191 ----
X  */
X+   malloc_sbrk_used += q->size;
X+   malloc_sbrk_unused -= q->size;
X+   malloc_bytes_in_free_list -= q->size;
X    return((char * )++q);
X***************
X*** 102,103 ****
X--- 198,200 ----
X
X+   if (!r) return;
X  /* move back to uncover the mem_chunk */
X***************
X*** 104,105 ****
X--- 201,205 ----
X    r--;          /* there it is! */
X+   malloc_sbrk_used -= r->size;
X+   malloc_sbrk_unused += r->size;
X+   malloc_bytes_in_free_list += r->size;
X  /*
X***************
X*** 158,160 ****
X    {             /* block too small, get new one */
X!   dst = q = (struct mem_chunk * )malloc(n);
X    if (q != NULL)
X--- 258,260 ----
X    {             /* block too small, get new one */
X!   dst = (long *) q = (struct mem_chunk * )malloc(n);
X    if (q != NULL)
X***************
X*** 190,191 ****
```

```
X  }
X-
X--- 290 ----
END_OF_FILE
if test 4440 -ne `wc -c <'memory.c.cdiff'`; then
    echo shar: \"'memory.c.cdiff'\" unpacked with wrong size!
fi
# end of 'memory.c.cdiff'
fi
if test -f 'pexecv.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'pexecv.c'\"
else
echo shar: Extracting \"'pexecv.c'\" \(3861 characters\)
sed "s/^X//" >'pexecv.c' <<'END_OF_FILE'
X#ifdef test
X#define testmain
X#include <stdio.h>
X#endif
X#include <osbind.h>
X#include <ctype.h>
X#include <time.h>
X
X#define alloca    __builtin_alloca
X
Xclock_t  _child_runtime;
Xchar    *xarg_pattern = "xArg=00000000";
Xchar    *pbp_pattern = "PBP=%ld";
X
Xint
XPexecv(int mode, const char *command, char *argv[], char *env[])
X{
X    extern char   **environ;
X    register char *new_env;
X    register int  env_len = 0;
X    register int  arg_len = 30; /* "PBP=%ld\0ARGV=" + 10 bytes slack */
X    register char *cp;
X    register char **ap, **ep;
X    register int  i;
X    int  old_parameter_passing;
X    char *tail;
X    char cmd[strlen(command)+1];
X    int  old_input, old_output, in, out;
X    clock_t  start_time;
X    short    argc;
X    struct {
X         long magic;
X         short    argc;
X         char **argv;
X         char *iovector;
X         long parent;
X    } xarg_vector;
X    extern long   _base;
X
```

```
X    strcpy(cmd, command);
X    unix_to_tos(cmd);
X    if(!env) env = environ;
X#ifdef test
X    printf("Pexecv(%d,%s,%06lx,%06lx)\n",mode,cmd,argv,env);
X    printf("calculating environment length ..."); fflush(stdout);
X#endif
X    ep = env;
X    if(ep) while(*ep) env_len += 1 + strlen(*ep++);
X#ifdef test
X    printf(" done (%d)\n", env_len);
X    printf("calculating argument length ..."); fflush(stdout);
X#endif
X    ap = argv;
X    argc = 0;
X    if(ap) while(*ap) {
X         arg_len += 1 + strlen(*ap++);
X         argc++;
X    }
X#ifdef test
X    printf(" done (%d, %d args)\n", arg_len, argc);
X    printf("copying environment ...\n");
X#endif
X    cp = new_env = alloca(env_len+arg_len+1);
X    ep = env;
X    if(ep) while(*ep && strncmp("ARGV=",*ep,5)) {
X         /* don't pass our own arguments */
X#ifdef test
X         printf("copy '%s'", *ep);
X#endif
X         if(strncmp(xarg_pattern,*ep,5) && strncmp(pbp_pattern,*ep,4)) {
X              strcpy(cp,*ep++);
X              cp += strlen(cp) + 1;
X#ifdef test
X              printf("\n");
X#endif
X         } else {
X              ep++;
X#ifdef test
X              printf(" skipped\n");
X#endif
X         }
X    }
X#ifdef test
X    printf("copying environment ... done\n");
X#endif
X    in = out = old_input = old_output = -10;
X    if(old_parameter_passing = isupper(argv[0][0])) {
X         /* This is for old compiler passes not recognizing xArg */
X         strcpy(cp,"ARGV=CCC?");
X         cp += 10;
X    }
X    tail = alloca(arg_len);
```

```
X    *tail = '\0';
X    ap = argv;
X    if(ap) while(1) {
X         strcpy(cp,*ap++);
X         if((*cp == '<' || *cp == '>') && !cp[1] && *ap) {
X              if(*cp == '>') {
X                   old_output = Fdup(1);
X                   Fforce(1, out = creat(*ap++, 0664));
X              } else {
X                   old_input = Fdup(0);
X                   Fforce(0, in = open(*ap++, 0));
X              }
X         } else if(old_parameter_passing) cp += strlen(cp) + 1;
X         if(!*ap) break;
X         strcat(tail," ");
X         strcat(tail,*ap);
X    }
X    *cp = '\0';
X    i = strlen(tail);
X    if(i > 126) i = 126;
X    *tail = i - 1;
X    tail[i] = '\0';
X    strcat(tail+1,"\r");
X    if(!old_parameter_passing) {
X         long tmp;
X#ifdef test
X         char *savepos;
X#endif
X
X         xarg_vector.magic = 0x78417267;
X         xarg_vector.iovector = (char *) 0;
X         xarg_vector.parent = _base;
X         xarg_vector.argv = argv;
X         xarg_vector.argc = argc;
X         strcpy(cp,xarg_pattern);
X#ifdef test
X         savepos = cp;
X#endif
X         cp += 13;
X         tmp = (long) &xarg_vector;
X         while(tmp) {
X              *--cp = "0123456789ABCDEF"[tmp & 0x0F];
X              tmp >>= 4;
X         }
X#ifdef test
X         printf("xArg Parameters\n");
X         printf("Environment variable = '%s'\n",savepos);
X         printf("\t   magic = '%4s'\n",&(xarg_vector.magic));
X         printf("\tiovector = %08lx\n",xarg_vector.iovector);
X         printf("\t  parent = %08lx\n",xarg_vector.parent);
X         printf("\t    argc = %d\n",xarg_vector.argc);
X         printf("\t    argv = %08lx\n",xarg_vector.argv);
X#endif
```

```
X    }
X    start_time = clock();
X    i = (Pexec(mode, cmd, tail, new_env));
X    _child_runtime += clock() - start_time;
X#ifdef test
X    printf("Pexec() returns %X = %d\n", i, i);
X#endif
X    if(old_output != -10) Fforce(1, old_output);
X    Fclose(old_output);
X    close(out);
X    if(old_input != -10) Fforce(0, old_input);
X    Fclose(old_input);
X    close(in);
X#ifdef test
X    fprintf(stderr, "Pexec() returns %X = %d\n", i, i);
X#endif
X    return(i);
X}
X
X#ifdef testmain
Xmain(int argc, char *argv[])
X{
X    printf("argc = %d\n", argc);
X    Pexecv(0, "a.out", argv, 0L);
X}
X#endif
END_OF_FILE
if test 3861 -ne `wc -c <'pexecv.c'`; then
    echo shar: \"'pexecv.c'\" unpacked with wrong size!
fi
# end of 'pexecv.c'
fi
if test -f 'signal.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'signal.c'\"
else
echo shar: Extracting \"'signal.c'\" \(5591 characters\)
sed "s/^X//" >'signal.c' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X * Use a signal stack when serving signals arrived in super mode (er) 14.8.89
X *
X * Added execute_in_user_mode and changed sig_array type, added some new
X * signals (SIGUSR, SIGHUP) (er)
X *
X * Adapted to ANSI-C (_do_signal renamed to raise) (er)
X * added return value and error checking to raise (er)
X *
X *   Signal routine - the signals are actually done by calling the _do_signal
X * routine - similiar to kill() I guess.
X *
X * $Header: signal.c,v 1.1 88/02/03 22:58:53 m68k Exp $
```

```
X *
X * $Log: signal.c,v $
X * Revision 1.1  88/02/03  22:58:53  m68k
X * Initial revision
X *
X */
X#include <signal.h>
X#include <errno.h>
X#include <osbind.h>
X#include <stdlib.h>
X
Xstatic void  stop(int);
Xextern void  _exit(int);
X
X/* ANSI C says that signal handlers have type (void (*)(int)) but even gcc */
X/* assumes that it's (int (*)(int)) */
X
X#define my_exit   ((sig_handler_t *) exit)
X#define my_quit   ((sig_handler_t *) _exit)
X#define my_stop   ((sig_handler_t *) stop)
X
Xstatic struct sigarray_str {
X    sig_handler_t *s_func; /* actual handler */
X    sig_handler_t *d_func; /* default handler */
X} sig_array[NSIG] = {
X    { SIG_DFL, SIG_IGN },  /* SIGNULL */
X    { SIG_DFL, SIG_IGN },  /* SIGALRM */
X    { SIG_DFL, my_exit },  /* SIGSEGV */
X    { SIG_DFL, my_exit },  /* SIGODD */
X    { SIG_DFL, my_exit },  /* SIGILL */
X    { SIG_DFL, SIG_IGN },  /* SIGZDIV */
X    { SIG_DFL, my_exit },  /* SIGCHK */
X    { SIG_DFL, my_exit },  /* SIGTRAP */
X    { SIG_DFL, my_exit },  /* SIGPRIV */
X    { SIG_DFL, SIG_IGN },  /* SIGTRACE */
X    { SIG_DFL, my_exit },  /* SIGINT */
X    { SIG_DFL, my_exit },  /* SIGQUIT */
X    { SIG_DFL, my_stop },  /* SIGTSTP */
X    { SIG_DFL, SIG_IGN },  /* SIGFPE */
X    { SIG_IGN, my_exit },  /* SIGHUP */
X    { SIG_DFL, my_quit },  /* SIGABRT */
X    { SIG_DFL, my_exit },  /* SIGTERM */
X    { SIG_DFL, SIG_IGN },  /* SIGUSR1 */
X    { SIG_DFL, SIG_IGN },  /* SIGUSR2 */
X};
X
X#undef my_exit
X#undef my_quit
X#undef my_stop
X
Xsig_handler_t *
Xsignal(int sig, sig_handler_t *func)
X{
```

```
X    sig_handler_t *oldfunc;
X
X    if (sig >= 0 && sig < NSIG) {
X         oldfunc = sig_array[sig].s_func;
X         sig_array[sig].s_func = func;
X    } else {
X         errno = EINVAL;
X         oldfunc = SIG_ERR;
X    }
X    return oldfunc;
X}
X
Xvolatile char     _sig_done;    /* needed for pause.c */
Xstatic long  _signal_stack[256];
Xstatic long  *const signal_stack = _signal_stack+256;
X
X/* We can't just switch the stack pointers to execute in usermode since
     */
X/* there are some programs which use their userstack as supervisorstack */
X/* too. We use a special signal stack which is reset to it's start if a */
X/* signal is received in supervisor mode. This is safe because the only */
X/* way to get back to the supervisor mode if you are in user mode would */
X/* use a call to the operating system (and so to the standard library), */
X/* which leeds to undefined behavior according to ANSI C. If we get the */
X/* signal in usermode we need not to worry about the stack.             */
Xstatic /* no inline */ void
Xexecute_in_usermode(sig_handler_t *func, int param)
X{
X    auto int  super_mode;   /* Avoid any problems with func by */
X    auto long super_stack;  /* putting these important vars on */
X    auto long user_stack;   /* the stack and not in registers. */
X
X    if (super_mode = Super(1)) {
X         asm ("movel sp, %0" : "=g" (super_stack));
X         asm ("movel usp, %0" : "=a" (user_stack));
X         asm ("andiw #0xDFFF, sr");  /* switch to the user mode */
X         asm ("movel %0, sp" :: "g" (signal_stack));
X    }
X    (void) (*func)(param); /* this is executed in user mode */
X    /* In the following cases we would never return to here: */
X    /*   - func terminates with exit or abort       */
X    /*   - func terminates with a longjmp      */
X    /* In both of the cases we don't need to know where the  */
X    /* userstack pointer was before the supervisormode was   */
X    /* entered since we don't want to get back there either. */
X    if (super_mode) {
X         Super(super_stack - 6);     /* this uses the user stack! */
X         /* Don't worry about deferred popping of param above when we */
X         /* have optimized this code with gcc -O. Just set the usp to */
X         /* the old value recorded at the start of this function. Any */
X         /* wrong setting of the ssp due to incorrect implementations */
X         /* of Super will be corrected by the following "unlk a6".    */
X         asm ("movel %0, usp" :: "a" (user_stack));
```

```
X    }
X}
X
Xstatic void
Xstop(int sig)
X{
X    Fwrite(2,23,"\r\nProgram stopped !!!\r\n");
X    system("-i");      /* interactively call the shell at *_shell_p */
X    Fwrite(1,2,"\33e");    /* some shells switch the cursor off on exit */
X}
X
Xint
Xraise(int sig)
X{
X    sig_handler_t *func;
X
X    if (sig >= 0 && sig < NSIG) {
X         /* we have to make sure that we call the signal-handler */
X         /* in user mode, to avoid problems on super stack later */
X         func = sig_array[sig].s_func;
X         if (func == SIG_DFL) func = sig_array[sig].d_func;
X         if (func != SIG_IGN) {
X              /* ANSI C requires either signal(sig,SIG_DFL) or an  */
X              /* "implementation defined blocking of the signal".  */
X              /* This blocking can be done by setting a flag which */
X              /* is cleared here after the call or in longjmp. But */
X              /* what about this scenario: signal(sig,abort) and   */
X              /* signal(SIGABRT,func) and func doing a longjmp ?   */
X              if(sig != SIGILL) sig_array[sig].s_func = SIG_DFL;
X              execute_in_usermode(func,sig);
X              _sig_done = 1;
X         }
X         /* else ignore it */
X         return(0);
X    }
X    /* invalid request for non existant signal */
X    errno = EINVAL;
X    return(-1);
X}
X
Xvoid
Xkill(int sig, int pid)
X{
X    extern const long  _base;
X
X    if(pid == _base) raise(sig);     /* we can't process kill requests */
X                     /* for other programs cause we do */
X                     /* not know their signal-handlers */
X}
END_OF_FILE
if test 5591 -ne `wc -c <'signal.c'`; then
    echo shar: \"'signal.c'\" unpacked with wrong size!
fi
```

```
# end of 'signal.c'
fi
if test -f 'stat.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'stat.c'\"
else
echo shar: Extracting \"'stat.c'\" \(4534 characters\)
sed "s/^X//" >'stat.c' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X * er: coupled this code with opendir/readdir/closedir, added fstat
X *
X * er: added inode stuff, file attributes (including executable), /dev/null
X *
X *  The only thing that doesn't get filled in at all is st_ino - it really
X * should be a unique number for each file - any ideas?
X *
X * $Header: stat.c,v 1.2 88/01/29 17:31:52 m68k Exp $
X *
X * $Log: stat.c,v $
X *
X * added fstat, inode numbers, more info about file modes and conversion of
X * unix names to tos syntax (er)
X *
X * 1.2 jrd
X *
X * Revision 1.1  88/01/29  17:31:52  m68k
X * Initial revision
X *
X */
X#include <types.h>
X#include <stat.h>
X#include <ctype.h>
X#include <errno.h>
X#include <osbind.h>
X#include <string.h>
X#include <param.h>
X#include <file.h>
X#include <dir.h>
X
Xstatic ino_t inode = 2;    /* FIX_ME: inodes are different in readdir */
XDIR *_current_open_directory = (DIR *) 0;
X
Xint
Xfstat(int fd, struct stat *st)
X{
X    if((fd >= 0) && (fd < N_HANDLES) && __handle_to_name[fd]) {
X        return(stat(__handle_to_name[fd], st));
X    } else {
X        errno = EBADF;
X        return -1;
X    }
```

```
X}
X
Xint
Xstat(p, st)
X    char     *p;
X    struct stat   *st;
X{
X    char     path[strlen(p)+1];
X    int      rval;
X    struct _dta   dtabuf;
X    extern long   _time();
X    int      fd;
X    short         magic;
X    short         dots;
X    int      len;
X
X    strcpy(path,p);
X    if(!strcmp(p,"/dev/null")) {
X         st->st_mode = S_IFBLK | 0666;
X         st->st_attr = FA_LABEL;
X         st->st_ino = 1;
X         goto specials;
X    }
X    unix_to_tos(path);
X    len = strlen(path);
X    if (path[len-1] == '\\' || len == 2 && path[1] == ':') {
X         if (len > 3 || len > 1 && path[1] != ':') {
X              path[--len] = 0;
X         } else {  /* root directory */
Xroot_dir:
X              st->st_mode = S_IFDIR | 0755;
X              st->st_attr = FA_DIR;
X              st->st_ino = 2;    /* In root dir '.' and '..' have the */
X                        /* same inode number; we should also */
X                        /* differentiate between the drives. */
Xspecials:
X              st->st_mtime = st->st_ctime = st->st_atime = 0;
X              goto fill_in;
X         }
X    }
X    if (!path) {
X         errno = EFAULT;
X         return -1;
X    }
X    if (index(path, '*') || index(path, '?')) {
X         errno = EPATH;
X         return -1;
X    }
X    if ((rval = Fsetdta(&dtabuf)) < 0) {
X         errno = -rval;
X         return -1;
X    }
X    dots = 0;
```

```
X    if (*path == '.') {
X         dots++;
X         if (len > 1) {
X              if (len == 2 && path[1] == '.') dots++;
X              else dots = 0;
X         }
X    }
X    if (dots) {
X         char actual_path[MAXPATHLEN];
X
X         Dgetpath(actual_path,0);
X         if (!*actual_path) goto root_dir;
X         if ((rval = Fsfirst("*.*", FA_DIR)) < 0) {
X              errno = -rval;
X              return -1;
X         }
X         if (dots == 2 && (rval = Fsnext()) < 0) {
X              errno = -rval;
X              return -1;
X         }
X         if (strcmp(path, dtabuf.dta_name)) {
X              errno = ENOENT;
X              return -1;
X         }
X    } else {
X         /* Optimization: if the file is already found in an open dir */
X         /*      use information collected there instead of Fsfirst() */
X         char *cp;
X         char buf[MAXPATHLEN+1];
X         DIR  *dp;
X         struct _dircontents    *dc;
X
X         strcpy(buf, path);
X         cp = rindex(buf, '\\');
X         if(cp) *cp++ = '\0';
X         else {
X              /* we are searching in '.' */
X              cp = path;
X              Dgetpath(buf, 0);
X         }
X         /* now we have the directory component in buf */
X         for(dp = _current_open_directory; dp; dp = dp->dd_next) {
X           if(!strcmp(dp->dd_name, buf)) {    /* WOW!, we got it */
X              for(dc = dp->dd_contents;dc;dc = dc->_d_next) {
X                   if(!strcmp(cp,dc->_d_entry)) {
X                        dtabuf.dta_attribute = dc->_d_attr;
X                        dtabuf.dta_date = dc->_d_date;
X                        dtabuf.dta_time = dc->_d_time;
X                        dtabuf.dta_size = dc->_d_size;
X                        goto got_dta;;
X                   }
X              }
X           }
```

```
X        }
X        if ((rval = Fsfirst(path, FA_SYSTEM|FA_HIDDEN|FA_DIR)) < 0) {
X             errno = -rval;
X             return -1;
X        }
X    }
Xgot_dta:
X    st->st_ino = ++inode;  /* should be able to do better then this */
X    st->st_mode = 0644 | (dtabuf.dta_attribute & FA_DIR ?
X                    S_IFDIR | 0111 : S_IFREG);
X    if(dtabuf.dta_attribute & FA_RDONLY) st->st_mode &= ~0222;
X    if(dtabuf.dta_attribute & FA_HIDDEN) st->st_mode &= ~0444;
X    st->st_mtime = st->st_ctime = st->st_atime =
X         _time(0L, (dtabuf.dta_date << 16) | dtabuf.dta_time);
X    st->st_attr = dtabuf.dta_attribute;
X    if (st->st_mode & S_IFREG) {
X         if ((fd = Fopen(path,0)) < 0) {
X             errno = -rval;
X             return -1;
X         }
X         Fread(fd,2,&magic);
X         Fclose(fd);
X         if (magic == 0x601A) st->st_mode |= 0111;
X         st->st_size = dtabuf.dta_size;
X         st->st_blocks = (dtabuf.dta_size + 1023) / 1024;
X    } else {
Xfill_in:
X         st->st_size = 1024;
X         st->st_blocks = 1;
X    }
X    if (*path && path[1] == ':')
X         st->st_dev = islower(*path) ? *path - 'a' : *path - 'A';
X    else
X         st->st_dev = Dgetdrv();
X    st->st_rdev = 0;
X    st->st_nlink = 1;
X    st->st_uid = 0;
X    st->st_gid = 0;
X    st->st_blksize = 1024;
X    return 0;
X}
END_OF_FILE
if test 4534 -ne `wc -c <'stat.c'`; then
    echo shar: \"'stat.c'\" unpacked with wrong size!
fi
# end of 'stat.c'
fi
if test -f 'system.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'system.c'\"
else
echo shar: Extracting \"'system.c'\" \(3327 characters\)
sed "s/^X//" >'system.c' <<'END_OF_FILE'
X/* 9.8.1989 (er)  changed return values of m_getenv() and g_getline(), to
```

```
X *           minimize program size (this file is contained in every
X *           program because of signal(SIGTSTP))
X *           registers d1-a5 are now saved before a system call (not
X *           necessary in m_getenv since Master will do this)
X */
X#include <osbind.h>
X#include <time.h>
X
Xstatic  long (*shell_p)();
Xextern  int  errno;
Xclock_t _sys_runtime;
Xchar    _shell_is_running = 0;
X
Xstatic void
Xread_shell(void)
X{
X    long *_shell_p = (long *) 0x4F6L;
X
X    shell_p = (long (*)())(*_shell_p);
X}
X
X/* This reads the _shell_p vector and tries to determine what kind */
X/* of shell we have running - !!! PLEASE ADD NEW MAGIC NUMBERS !!! */
Xstatic int
Xinit_system(void)
X{
X    int  ret;
X
X    Supexec(read_shell);
X    if(!(long)shell_p) return(0);
X    ret = 1;
X    if((*((long *)((long) shell_p)-10)) == 0x00420135L))
X         ret = 2; /* Gulam */
X    if((*((long *)((long) shell_p)-4)) == 0xCDCEC5D2L) || /* Master 5.0 */
X       (*((long *)((long) shell_p)-8)) == 0x4D415354L))   /* Master 5.4 */
X         ret = 3; /* Master */
X    return(ret);
X}
X
X/* this function executes the given command cmd by the shell at _shell_p   */
X/* if cmd == NULL only the presence of a shell (and it's type) is reported */
X/* when we want an interactive shell (cmd == "-i") then substitute this to */
X/* "" if the shell is Gulam - we could also Pexec("$SHELL") if there is no */
X/* shell at _shell_p */
Xint
Xsystem(const char *cmd)
X{
X    int  ret;
X    clock_t  start_time;
X
X    switch(ret = init_system()) {
X         case 0 : return(0);
X         case 2 : if (!strcmp(cmd,"-i")) cmd = "ue";
```

```
X        default: if (cmd) {
X                 start_time = clock();
X                 _shell_is_running = 1;
X                 asm("moveml d1-a5, sp@-");
X                 asm("movel %0, sp@-" :: "g" (cmd));
X                 (void) (shell_p[0])();
X                 asm("addql #4, sp");
X                 asm("moveml sp@+, d1-a5");
X                 asm volatile ("movel d0, %0" : "=g" (ret));
X                 _shell_is_running = 0;
X                 _sys_runtime += clock() - start_time;
X                 if(ret < 0) errno = -ret;
X             }
X             return(ret);
X    }
X    /*NOTREACHED*/
X}
X
X/* special system call if the shell is Master 5.x */
X/* we ask for the expansion of the shell variable var, if Master is not     */
X/* running we return with the expansion as environment var - we could also  */
X/* get shell vars from other shells with output redirection like this one:  */
X/* system("echo $var > var.tmp") and then read this file if we got no error */
Xchar *
Xm_getenv(const char *var)
X{
X    extern long   getenv();
X    char *result;
X
X    if(init_system() == 3) {
X         _shell_is_running = 1;
X         result = (char *) (shell_p[4])(1L,var);
X         _shell_is_running = 0;
X    } else result = "";             /* getenv(var); */
X    return(result);
X}
X
X/* special system call if the shell is Gulam */
X/* we ask for a line of input provided by the shell (with all usual   */
X/* editing), if Gulam is not running we simply read a line from stdin */
X/* we could also use system("ask \"\" tmpvar") and then get the value */
X/* of $tmpvar with m_getenv("tmpvar") if the shell is Master */
Xvoid
Xg_getline(char *line)
X{
X    void (*func)(void);
X
X    if(!line) return;
X    if(init_system() != 2) *line = '\0'; /* read(0,line,256); */
X    else {
X         _shell_is_running = 1;
X         func = (void (*)(void))&(shell_p[-6]);
X         asm("moveml d1-a5, sp@-");
```

```
X         asm("movel %0, sp@-" :: "g" (line));
X         (*func)();    /* the argument is already on the stack */
X         asm("addql #4, sp");
X         asm("moveml sp@+, d1-a5");
X         _shell_is_running = 0;
X    }
X}
END_OF_FILE
if test 3327 -ne `wc -c <'system.c'`; then
    echo shar: \"'system.c'\" unpacked with wrong size!
fi
# end of 'system.c'
fi
if test -f 'timer.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'timer.c'\"
else
echo shar: Extracting \"'timer.c'\" \(3923 characters\)
sed "s/^X//" >'timer.c' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X *   This routine uses the vertical blank interrupt queue to get it's timing
X * done.  This means that it probably depends on the resolution of the screen.
X *   If there is no room in the vbl queue then -1 is returned.  Note that
X * before the program terminates, this routine must be taken out of the
X * queue - to do this, _unalarm() is called from exit().  If alarm() is
X * not used in the program, the _unalarm() routine is a null one.
X *   This routine works by messing with the return pc (and sr) on the stack
X * when it gets called from the vbl interupt routine.  Also, the routine
X * checks to make sure a trap is not in progress by checking the __in_trap
X * variable (which *must* be set to one before a trap is done, and cleared
X * afer by the programmer - this is normally handled by the __*bios routines).
X * All this weirdness is done to ensure that the os is not disturbed while
X * it is doing stuff - therefor it is safe to do a longjmp() inside a signal
X * handling routine.
X *
X * $Header: alarm.c,v 1.1 88/02/03 22:37:51 m68k Exp $
X *
X * added SIGHUP, _init_timer and _exit_timer (called from crt0.s) (er)
X * renamed to timer.c and included clock()-function (er)
X *
X * $Log: alarm.c,v $
X * Revision 1.1  88/02/03  22:37:51  m68k
X * Initial revision
X *
X */
X#include <osbind.h>
X#include <sysvars.h>
X#include <signal.h>
X#include <time.h>
X
X#define  HZ       (70)     /* how often we get called a second */
```

```
X#define  CLK_VEC       0x100         /* the clock vector */
X
Xtypedef long *long_ptr;
X
Xextern volatile int   _received_signal;  /* in alrm_nasty.s */
Xextern char       _shell_is_running; /* in system.c */
Xextern const long _base;
Xextern const long_ptr  _act_pd;
X
Xstatic int   alarm_timer();
Xextern int   _exit_timer();
Xextern int   _init_timer();
Xstatic int   remove_timer();
Xstatic int   install_timer();
X
Xclock_t  _start_time = 0L;
X
Xstatic int
X_clock(void)
X{
X    return *((clock_t *)_hz_200);
X}
X
Xclock_t
Xclock(void)
X{
X    return (Supexec(_clock) - _start_time);
X}
X
Xstatic int   carrier_was_on = 0;
Xstatic long  alarmtime = 0;
Xstatic clock_t    trigger = (clock_t) 0;
X
Xint
Xalarm(int n)
X{
X    long     oldticks;
X
X    trigger = (clock_t) 0;               /* block alarms now */
X    oldticks = alarmtime;
X    alarmtime = n;                       /* that's the new one */
X    if(n) trigger = Supexec(_clock) + n * CLK_TCK; /* and go */
X    return oldticks;
X}
X
Xint
X_init_timer(void)
X{
X    return Supexec(install_timer);
X}
X
Xint
X_exit_timer(void)
```

```
X{
X    return Supexec(remove_timer);
X}
X
X/*
X * This routine should be as small as possible as it is called lots of times
X * (There isn't really an argument - it's just there so we can get the
X * address of the stack)
X * If we just received another signal don't do anything - just wait until it's
X * being processed. (er)
X */
Xasm(".ascii  \"XBRAGNUC\"");    /* protocoll */
Xasm(".long   0");      /* old vector == no vector */
X
Xstatic int
Xalarm_timer(int arg)
X{
X    if(_shell_is_running || *_act_pd != _base || _received_signal)
X         return 0; /* too dangerous to process signals */
X    if (trigger && *((clock_t *)_hz_200) > trigger) {
X         /* rrriiinnnnggggggggg! */
X         alarmtime = trigger = 0;
X         _received_signal = SIGALRM;
X         _alrm_nasty(&arg);
X    } else {
X         int carrier_is_off = (*(char *) 0x00FFFA01L) & 2;
X
X         if (carrier_was_on && carrier_is_off) {
X              carrier_was_on = 0;
X              _received_signal = SIGHUP;
X              _alrm_nasty(&arg);
X         } else carrier_was_on = !carrier_is_off;
X    }
X    return 0;
X}
X
X/* These routines must be called in super mode */
Xstatic  int
Xremove_timer(void)
X{
X    char **func;
X    int  i;
X
X    trigger = (clock_t) 0;  /* no more alarms now */
X    func = *_vblqueue;
X    i = *nvblq;
X    while (i--)
X         if (*func == (char *) alarm_timer) {
X              *func = (char *) 0;
X              return 0;
X         } else func++;
X    return 1;
X}
```

```
X
Xstatic  int
Xinstall_timer(void)
X{
X    char **func;
X    int  i;
X
X    func = *_vblqueue;
X    i = *nvblq;
X    while (i--)
X         if (!*func) {
X              *func = (char *) alarm_timer;
X              return 0;
X         } else func++;
X    return 1;
X}
END_OF_FILE
if test 3923 -ne `wc -c <'timer.c'`; then
    echo shar: \"'timer.c'\" unpacked with wrong size!
fi
# end of 'timer.c'
fi
echo shar: End of archive 4 \(of 7\).
cp /dev/null ark4isdone
MISSING=""
for I in 1 2 3 4 5 6 7 ; do
    if test ! -f ark${I}isdone ; then
     MISSING="${MISSING} ${I}"
    fi
done
if test "${MISSING}" = "" ; then
    echo You have unpacked all 7 archives.
    rm -f ark[1-9]isdone
else
    echo You still need to unpack the following archives:
    echo "        " ${MISSING}
fi
##  End of shell archive.
exit 0
```

```
From 9066127068329923216
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,88d2e92bc0307a6,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-08-19 10:53:46 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!csd4.csd.uwm.edu!cs.utexas.edu!
sun-barr!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i076: gcclib2 -- Patches to GCC library  part05/07
Keywords: shar, C
Message-ID: <34138@grapevine.uucp>
```

Date: 19 Aug 89 17:53:46 GMT
Sender: news@grapevine.uucp
Lines: 921
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: uunet.UU.NET!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 2, Issue 76
Archive-name: gcclib2/part05

```
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of archive 5 (of 7)."
# Contents:  crt0.s read.c
# Wrapped by roeder@sbsvax on Wed Aug 16 22:03:03 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'crt0.s' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'crt0.s'\"
else
echo shar: Extracting \"'crt0.s'\" \(11605 characters\)
sed "s/^X//" >'crt0.s' <<'END_OF_FILE'
X|
X|       Cross Development System for Atari ST
X|      Copyright (c) 1988, Memorial University of Newfoundland
X|
X|  C start-up routine.  Has several tasks to do:
X|   release unneeded memory to the system
X|   set up stack
X|   set up argc, argv, and envp
X|   set up an address trap handler for bsr.l instructions
X|  Also defines _exit() so the address error handler can be restored
X|
X| $Header: /chandra/Gnu/atari/cross-lib/RCS/crt0.s,v 1.3 89/03/10 20:49:42 bammi
Exp $
X| $Header: crt0.s,v 1.3 88/02/03 20:23:26 m68k Exp $
X|
X| Revision 1.9  89/08/15  er
X| Merged local version and version from terminator together
X| completely removed __in_trap flag (it's no longer needed - cf. signal.c)
X|
X| FIXME: all this revision number confusion should be cleaned up
X|
X| Revision 1.8  89/08/07  er
X| Added _init_timer and _exit_timer to do signal processing for SIGALRM/SIGHUP
X| Added argc/argv for accessories (just dummies)
X|
X| $Log: crt0.s,v $
X| Revision 1.3  89/03/10  20:49:42  bammi
```

```
X| backed out Fforce(2,Fdup(1)) due to problems with redirections.
X| Revision 1.7  89/08/01  sb
X|     New argc/argv parsing for the commandline. Old code didn't recognize
X| multiple blanks between arguments and supplied each blank as "" argument.
X|     Accessories need their own stack.
X| Sample code in the main-module:
X|   #define STACKSIZE 8000L
X|   char my_private_acc_stack[STACKSIZE];
X|   extern long _heapbase = (long) my_private_acc_stack;
X|   extern long _stksize  = STACKSIZE;
X|
X| Revision 1.2  89/03/08  22:20:51  bammi
X| Many hacks etc. Version as distributed with Gcc V1.34
X|
X| Revision 1.1  89/03/08  21:57:52  bammi
X| Initial revision
X|
X| Revision 1.6  89/07/19  er/sb/AD
X| __in_trap flags removed from early trap calls since alarm can't yet be there
X| added stuff from sb (__app and accessory support) and AD (malloc changes)
X|  Revision 1.2.1  89/02/06  AD (Atze Dijkstra)
X|  Changed malloc/stack mgt so stack is on top of heap
X|  and they will grow to each other.
X|  These will only be if __stksize == -1L
X|  See also memory.c
X|
X| Revision 1.5  89/07/06  er
X| changed to xArg parameter passing
X|
X| Revision 1.4  89/06/02  er
X| Added support for extended argument passing via PBP/ARGV
X| Installs trap handler for etv_term to restore alarm-routine and address-
X| error-handler. This handler catches also ^C pressed during TOS-i/o.
X| Added data_start and _base
X| TODO: _exit could be removed now from this code so the user can redefine it
X|
X| $Log: crt0.s,v $
X| Revision 1.3  88/02/03  20:23:26  m68k
X| Added __in_trap flag - see atari/alarm()
X|
X| Revision 1.2  88/01/29  17:24:37  m68k
X| Re-wrote the argument parsing stuff
X| Calls exit() instead of _exit()
X| uses Setvec() trap instead of Super()/mov/Super() to change the ae trap
X| stack now has initial size of 8k (up from 2)
X|
X|
X    .globl  _data_start
X    .globl   __start
X    .globl   __exit
X    .globl   __base
X    .globl   _environ
X    .globl   __heapbase        | AD
```

```
X    .globl   _main
X    .globl   _errno
X    .globl   __app
X    .globl   __init_timer
X    .globl   __exit_timer
X
X    .data
X
X    .comm    _data_start,4
X    .comm    __base,4
X    .comm    __stksize,4
X    .comm    _environ,4
X    .comm    __heapbase,4       | AD
X    .comm    _errno,4
X    .comm    __act_pd,4
X        .comm   __app,4             | ++sb  acc or prg ?
X
X    .text
X
X__start:
X    cmpl #0, a7             | ++sb
X    bne  L0_1
X    movl a0, a5
X    bra  L0_2
XL0_1:    movl a7@(4), a5
XL0_2:    movl a5, __base
X
X    movl #1, __app
X    tstl a5@(36)            | sb in accessories the father is NULL
X    bne  L25
X    clrl __app              | please see demo-source for acc's!
X    movl __heapbase, sp         | ++ sb acc NEED his own stack!
X    addl __stksize, sp      | ++ sb
X    movl #1, d3             | argc = 1    (er)
X    lea  acc_argv, a2       | argv = acc_argv = { "", 0L }
X|   lea  arg0, a0
X|   movl a0, a2@            | argv[0] = ""
X    bra  L31_1              | ++ sb (no envp/argv stuff for acc's)
X
XL25:
X    | set up envp[] and argv[] stuff
X    movl a5@(0x18), a1      | envp = basepage->bss_address
X    addl a5@(0x1c), a1      |    + basepage->bss_size;
X    movl a1, _environ       | _environ = envp;
X    movl a5@(0x2c), a2      | ep = basepage->envp;
XL0: tstb a2@           | while (*ep) {
X    beq  L3            |
X    movl a2, a1@+      |    *envp++ = ep;
XL2: tstb a2@+          |    while (*ep++)
X    bne  L2            |         ;
X    bra  L0            | }
XL3: clrl a1@+          | *envp++ = (char *) 0;
X
```

```
X    movl _environ, a3       | now we search for the xArg env-var
XL1: tstl a3@           | for (s = environ ; *s ; s++)
X    beq  L7            |
X    movl a3@+, a4      |    if (strncmp(*s, "xArg=", 5))
X    cmpb #0x78, a4@+       |    'x'
X    bne  L1            |
X    cmpb #0x41, a4@+       |    'A'
X    bne  L1            |
X    cmpb #0x72, a4@+       |    'r'
X    bne  L1            |
X    cmpb #0x67, a4@+       |    'g'
X    bne  L1            |
X    cmpb #0x3d, a4@+       |    '='
X    bne  L1            |
X
X    subl a2, a2
X    lea  _conv,a0      | here we do a hex to binary conversion
XL14:
X    movl a2, d3
X    lsll #4, d3
X    movl d3, a2
X    moveq    #15,d0
XL15:
X    moveb    a0@(d0:l),d3
X    cmpb a4@,d3
X    jeq  L16
X    subql    #1,d0
X    jne  L15
XL16:
X    movl a2, d3
X    orl  d0, d3
X    movl d3, a2
X    addql    #1,a4
X    tstb a4@
X    bne  L14
X    cmpl #0x78417267,a2@        | is this a valid xArg-structure ?
X    bne  L7
X    movel    a2@(14),d3
X    cmpl a5@(36),d3         | does it come from our father process ?
X    bne  L7
X    movew    a2@(4),d3      | here is argc (as word)
X    extl d3
X    movel    a2@(6),a2      | and argv
X    movl a1, a0
X    bra  L6
X_conv:
X    .ascii "0123456789ABCDEF"
XL7:                    | /* a1: envp -> argbuf */
X    movl a1, a0             | argv = argbuf
X    addl #128, a0      |     + sizeof(basepage->cmdline);
X    clrw a0@+          | *((short)argv)++ = 0;
X    movl a1, d1             | argstart = argbuf;
X    movl a0, a2             | saveargv = argv;
```

```
X|   movew    #1, d3             | argc = 1;
X    movel    #1, d3             | .. jrd
X    movl #arg0, a0@+        | *argv++ = "";
X    movl #0x80, a3     | cmdline = basepage->cmdline;
X    addl a5, a3             |
X| new parsing of cmdline ... sb (1 aug 89)
     | sb
X    clrw d1            | in_arg = FALSE;      | sb
X    movb a3@+, d0      | count = *cmdline++;       | sb
XL8: subqb    #1, d0             | while (--count > 0) {
     | sb
X    blt  L9            |                  | sb
X    tstb a3@           |   if (!*cmdline)     | sb
X    beq  L9            |    break;            | sb
X    cmpb #0x20,a3@     |   } else if (*cmdline==' ') { | sb
X    bne  L12           |                  | sb
X    clrw d1            |       in_arg = FALSE;    | sb
X    clrb a3@+          |       *cmdline = '\0';   | sb
X    bra  L8            |   } else if (!in_arg) {   | sb
XL12:    tstw d1             |                  | sb
X    bne  L11           |                  | sb
X    movw #1, d1             |       in_arg = TRUE; | sb
X    addql    #1, d3             |       argc++;        | sb
X    movl a3, a0@+      |       *argv++ = cmdline; | sb
XL11:    addql    #1, a3             |   } cmdline++;
     | sb
X    bra  L8            | }                | sb
XL9:                                    | sb
X    | remove trailing CR                          | sb
X    clrb a3@           | *cmdline = '\0';     | sb
X    clrl a0@+          | *argv++ = NULL;      | sb
X
XL6:
X    cmpl #0xFFFFFFFF, __stksize | if ( __stksize == -1 )
     | AD
X    bne  L30           |                  | AD
X    movl a0, __heapbase         | __heapbase = argv
     | AD
X    movl sp, a3             | __stksize = ( sp - argv -12K )| AD
X    subl a0, a3                             | AD
X    subl #0x3000, a3        | no checks !!              | AD
X    movl a3, __stksize                      | AD
X    addl a0, a3             | reserve = __stksize + argv
     | AD
X    subl a5, a3             |        - __base | AD
X    bra  L31                                | AD
XL30:                                        | AD
X    movl #0, __heapbase         | __heapbase = 0       | AD
X    movl a0, a3             | reserve = argv - __base;
X    subl a5, a3
X    tstl __stksize     | if (!__stksize)
X    bne  L13           |
X    movl #0x2000, __stksize |    __stksize = 8 * 1024;
```

```
XL13:    addl __stksize, a3      | reserve += __stksize;
XL31:    movl a3, d1
X    addl a5, d1             | compute stack top
X    andl #-2, d1            | even boundary
X    movl d1, sp             | set up user stack
X
X    movl a3, sp@-
X    movl a5, sp@-
X    clrw sp@-
X    movw #0x4a, sp@-
X    trap #1             | release excess storage
X    addl #12, sp
X
XL31_1:  clrl sp@-           | act_pd is needed for etv_term
X    movw #0x20, sp@-
X    trap #1
X    movl #0x4F2, a0         | _sysbase
X    movl a0@, a3            | Super() trashes a0-a1/d0-d1
X    movl d0, sp@(2)
X    trap #1
X    addql    #6, sp
X    movl a3@(24), a4        | os_gendat
X    cmpl #0x11201985, a4
X    beq  L24
X    cmpl #0x02061986, a4
X    beq  L24
X    cmpl #0x04241986, a4
X    beq  L24
X    movl a3@(40), __act_pd  | it's in the header for newer versions
X    bra  L17
XL24:    movl #0x602C, __act_pd  | default in old TOS versions
X
XL17:    movl _environ, sp@-         | push saveenvp
X    movl a2, sp@-       | push saveargv
X|   movw d3, sp@-       | push argc
X    movel    d3,sp@-            | .. jrd
X
X    subl a6, a6             | clear link reg - why I don't know
X
X    | here we set up a trap handler for address errors - this is used in
X    | doing long branches and long bsrs (see comments in Leat()).
X
X    movl #aet, sp@-     | set address error vector to our handler
X    movew    #0x3, sp@-     |  was 0xc bug! thanks to sarnila@tukki.jyu.fi
X    movew    #0x5, sp@-     |  for finding and reporting it.
X    trap #13
X    addql    #0x8, sp
X    movl d0, old_aet    | save old trap handler
X
X    movl __app, d0| no terminate handler for accessories
X    beq  L32
X    movl #term, sp@-    | install our terminate handler
X    bra  L33
```

```
XL32:    movl #-1, sp@- | for accessories: just ask for old handler
XL33:    movew    #0x102, sp@-
X    movew    #0x5, sp@-
X    trap #13
X    addql    #0x8, sp
X    movl d0, old_term  | save old trap handler
X
X|   jsr  _main
X    jsr  __init_timer
X    jsr  __main        | -- jrd.  call the standard wrapper fun
X    movew    d0, sp@-
X    jsr  _exit         | _exit calls __exit
X    | Not reached
X
X|
X| this handler is installed to insure that our signal handling will be removed
X| when it's time to go
X| it does also some signal-processing (^C pressed during TOS-i/o)
X|
X    .ascii   "XBRA"
X    .ascii   "GNUC"
Xold_term:
X    .long    0
Xterm:
X    moveml   d0-d1/a0, sp@-
X    movl __act_pd, a0  | is the actual running process this one ?
X    movl a0@, d0
X    movl __base, d1
X    cmpl d0, d1
X    beq  L23
X    moveml   sp@+, a0/d0-d1     | this signal was not meant for us
X    rts
XL23:    moveml   sp@+, a0/d0-d1
X    movew    a6@(8), d0
X    cmpw #-32, d0  | ^C pressed or normal exit ?
X    bne  L22
X| during the following call to raise() we should better not call GEMDOS,
X| because GEMDOS should not be called with its own super stack in effect
X    moveml   d1-d7/a0-a5, sp@-
X    movel    #10, sp@- | raise(SIGINT)
X    jsr  _raise
X    addql    #4, sp
X    moveml   sp@+, a0-a5/d1-d7
X|
X| ignore ^C now
X| this trick depends on the following facts:
X| 1. the TOS-i/o routine calls Pterm(-32) directly as a function and removes
X|    the parameter (-32) if this routine eventually comes back
X| 2. Pterm itself calls the logical vector 0x102 (etv_term) via assembler
X|    and so the register a6 points to the params of Pterm
X| 3. after the 'unlk a6', we can return to the caller of Pterm just with 'rts'
X|
X    unlk a6
```

```
X    rts
XL22:    movl old_aet, sp@- | restore old address trap before we exit
X    movew    #0x3, sp@-
X    movew    #0x5, sp@-
X    trap #13
X    addql    #0x8, sp
X    movl old_term, sp@-    | restore old exit trap before we exit
X    movew    #0x102, sp@-
X    movew    #0x5, sp@-
X    trap #13
X    addql    #0x8, sp
X    jmp  __exit_timer
X__exit:
X    jsr  L22      | remove our stuff
X|   movew    sp@(0x4), sp@-     | .. jrd
X    movel    sp@(4),d0
X    movew    d0,sp@-
X    movew    #0x4c, sp@-
X    trap #1
X    | The End.
X
X|  This is a trap handler for address errors.  It is used in doing long
X| branches and long bsrs.  It looks to see if a long branch (0x6?ff) caused
X| the trap and if the destination is valid (not an odd address).  If so
X| the branch is taken and we return, otherwise, we call the old address
X| trapper.
X    .ascii   "XBRA"
X    .ascii   "GNUC"
Xold_aet:
X    .long    0
Xaet:
X    | d0, and a1 used 0x8080/101
X    | exp_frame is distance from top of stack to exception frame
X    Lexp_frame = 8              | 2 * sizeof(register)
X    moveml   #0x8080, sp@-      | save registers used
X    movw sp@(Lexp_frame+6), d0  | the bad instruction
X    andw #0xf0ff, d0
X    cmpw #0x60ff, d0        | is a b?? instruction?
X    jne  L20           | nop, call old ae trap
X
X    movl sp@(Lexp_frame+10), a0 | the pc (currently bewtixt branch)
X    addql    #1, a0             | increment to long offset
X    addl a0@, a0           | calculate new pc
X    movl a0, d0
X    andb #1, d0             | are we still an odd address?
X    jne  L20           | yep, get out.
X
X    movl a0, sp@(Lexp_frame+10) | put new pc into the stack frame
X    cmpw #0x61ff, sp@(Lexp_frame+6)  | if this is a bsrl
X    jne  L21           |  we need to fix up the return
X    movl usp, a0            |  address on the user stack
X    addql    #4, a0@
XL21:
```

```
X    moveml    sp@+, #0x0101      | restore regs
X    addql     #8,sp              | pop part of stack frame
X    rte
X
XL20:
X    moveml    sp@+, #0x0101
X    movl old_aet, sp@(-4)   | done this way so no regs get mussed up
X    jmp  sp@(-4)
X
X.data
Xacc_argv:                   | ANSI Draft: argv[0][0] shall be the
X    .long     arg0          | null character if the program name is
Xarg0:    .long     0             | not available in the host environment
END_OF_FILE
if test 11605 -ne `wc -c <'crt0.s'`; then
    echo shar: \"'crt0.s'\" unpacked with wrong size!
fi
# end of 'crt0.s'
fi
if test -f 'read.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'read.c'\"
else
echo shar: Extracting \"'read.c'\" \(8951 characters\)
sed "s/^X//" >'read.c' <<'END_OF_FILE'
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X * $Header: read.c,v 1.2 88/02/03 22:56:30 m68k Exp $
X *
X * $Log: read.c,v $
X *
X * 1.5       er
X *
X * Added /dev/null dummy handle.
X * Fix it : When adding 'continue' after raise(SIGINT) and raise(SIGQUIT), get
X *      unexpected results in Emacs (perhaps problems with EOF in fgetc()).
X * Moved __col_pos to write.c (it's always included because of fflush in exit)
X *
X * 1.4       er
X *
X * Revision 1.4    89/01/15
X * Added support for function keys (they give an Esc-sequence with the
X * scan-code as parameter), META-key (Alternate sets the highest bit) and
X * O_NDELAY mode (no waiting if there is no available input). META-translation
X * is controled via the META-bit in __ttymode (set by ioctl()).
X *
X * 1.3       jrd
X *
X * Revision 1.2  88/02/03  22:56:30  m68k
X * Added unix like tty driver stuff
X *
X * Revision 1.1  88/01/29  17:31:39  m68k
```

```
X * Initial revision
X *
X */
X#include <osbind.h>
X#include <ioctl.h>
X#include <file.h>
X#include <errno.h>
X#include <signal.h>
X#include "tchars.h"
X
X/* set this frob if using Bconin to get char.  Leave it off using Fread */
X#define USING_BCONIN  0
X#define META_MASK 0x80
X#define DEL       0x7f
X
X#define  iswhite(c)    ((c) == ' ' || (c) == '\t')
X#define  isvisable(c)  ((c) >= ' ' && (c) < DEL)
X#define  echochar(c)   if (__ttymode & ECHO) (void) _echochar(c, fd); else
X#define  delchar(n)    if (__ttymode & ECHO) _delchar(n, fd); else
X
X# ifndef NULL
X#  define    NULL 0
X# endif /* NULL */
X
Xstatic int str_length(unsigned char * p, int n);
X
Xextern int   __col_pos;
Xstatic int   start_col;
Xstatic unsigned char  *thebuf;
X
Xstatic  int  _echochar();
Xstatic  int  str_length();
Xstatic  void _delchar();
X
Xunion key {
X    int  i;
X    struct {
X         char shift_state;
X         char scan_code;
X         char filler;
X         unsigned char ascii_code;
X    } k;
X};
X
X/* possible shift_state's */
X#define  SHIFT_RIGHT   1
X#define  SHIFT_LEFT    2
X#define SHIFT     (SHIFT_LEFT|SHIFT_RIGHT)
X#define  CONTROL       4
X#define  ALTERNATE 8
X#define  CAPS_LOCK 16
X#define  MOUSE_RIGHT   32
X#define  MOUSE_LEFT    64
```

```
X#define  MOUSE    (MOUSE_LEFT|MOUSE_RIGHT)
X
Xstatic   enum { Esc, Number, Key }   _fn_state;
Xstatic   char _fn_key;
X
Xint
Xread(int fd, unsigned char *buf, int nbytes)
X{
X#ifdef DEBUG
X  char dbgbuf[64];
X#endif
X
X  int    rval;
X  int    cnt = 0;
X  unsigned char   *p = buf;
X  union key  ch;
X
X  if (fd == 1234 || nbytes <= 0) return(0);   /* /dev/null */
X  if ((fd >= 0) && (!isatty(fd)))
X    {
X    if ((rval = Fread(fd, nbytes, buf)) < 0)
X         {
X         errno = -rval;
X         rval = -1;
X         }
X#ifdef DEBUG
X    sprintf(dbgbuf, "read(%d, %X, %d)->%d\r\n", fd, buf, nbytes, rval);
X    dbgstr(dbgbuf);
X#endif
X    return rval;
X        }
X  if ((__handle_stat[fd] & O_NDELAY) && _fn_state == Esc && !Cconis()) {
X    errno = EAGAIN;
X    return -1;
X  }
X  thebuf = buf;
X  start_col = __col_pos;
X  while (1) {
X        switch (_fn_state) {
X                case Number :
X             *p = '#';
X             _fn_state = Key;
X             break;
X        case Key :
X             *p = _fn_key;
X             _fn_state = Esc;
X             break;
X        case Esc :
X             ch.i = console_read_byte(fd);
X             if((__ttymode & META) && !ch.k.shift_state) {
X                  struct sgttyb sg;
X
X                  ioctl(fd, TIOCGETP, &sg);
```

```
X                ioctl(fd, TIOCSETN, &sg);
X                ch.k.shift_state = Kbshift(-1);
X            }
X            if((__ttymode & META) &&
X                (ch.k.ascii_code & META_MASK)) {
X                /* these values are on the german keyboard */
X                switch(ch.k.ascii_code) {
X                    case 0x81 :
X                        ch.k.ascii_code = '@'; break;
X                    case 0x84 :
X                        ch.k.ascii_code = ']'; break;
X                    case 0x8E :
X                        ch.k.ascii_code = '}'; break;
X                    case 0x94 :
X                        ch.k.ascii_code = '['; break;
X                    case 0x99 :
X                        ch.k.ascii_code = '{'; break;
X                    case 0x9A :
X                        ch.k.ascii_code = '\\'; break;
X                }
X            }
X            if (ch.k.scan_code == 0x72 && ch.k.ascii_code == '\r')
X                ch.k.ascii_code = '\n';
X            else
X            /* 3B = F1, 72 = Enter, 53 = Delete, 60 = '<' */
X            if ((ch.k.scan_code >= 0x3B && ch.k.scan_code <= 0x72
X              && ch.k.scan_code != 0x53 && ch.k.scan_code != 0x60))
X            {
X                if (ch.k.ascii_code) _fn_key = ch.k.ascii_code;
X                else _fn_key = ch.k.scan_code;
X                ch.k.ascii_code = 0x1B; /* ESC */
X                _fn_state = Number;
X            }
X            *p = ch.k.ascii_code;
X            /* Alternate as META-key sets the highest bit */
X            if ((__ttymode & META) &&
X                (ch.k.shift_state & ALTERNATE)) {
X                if (_fn_state != Esc) _fn_key |= META_MASK;
X                else {
X                    if (ch.k.shift_state & SHIFT)
X            ch.k.ascii_code = __keytab->shift[ch.k.scan_code];
X                    else if (ch.k.shift_state & CAPS_LOCK)
X            ch.k.ascii_code = __keytab->capslock[ch.k.scan_code];
X                    else
X            ch.k.ascii_code = __keytab->unshift[ch.k.scan_code];
X                    if (ch.k.shift_state & CONTROL)
X                        ch.k.ascii_code &= 0x3F;
X                    *p = ch.k.ascii_code | META_MASK;
X                }
X            }
X            break;
X        }
X    if (__ttymode & RAW)
```

```
X           {
X        if (__ttymode & ECHO)
X                {
X             console_write_byte(fd, *p);
X             if (*p == '\r')
X                     __col_pos = 0;
X                 else
X             if (isvisable(*p))
X                  __col_pos++;
X             }
X        if (++cnt >= nbytes || !(short)console_status(fd))
X             return cnt;
X        p++;
X        continue;
X        }
X    if ((__ttymode & CRMOD) && *p == '\r')
X         *p = '\n';
X    if (*p == __tchars[TC_INTRC])
X         {
X         /* Do the bsd thing here, i.e. flush buffers
X          * and continue to read after the interupt
X          */
X         echochar(*p);
X         p = buf;
X         cnt = 0;
X         raise(SIGINT);
X         }
X        else
X    if (*p == __tchars[TC_QUITC])
X         {
X         echochar(*p);
X         p = buf;
X         cnt = 0;
X         raise(SIGQUIT);
X         }
X            else
X    if (*p == __tchars[TC_SUSPC])
X         {
X         echochar(*p);
X         p = buf;
X         cnt = 0;
X         raise(SIGTSTP);
X         continue;
X         }
X    if (__ttymode & CBREAK)
X         {
X         if (*p == __tchars[TC_LNEXTC])
X              *p = console_read_byte(fd);
X         if (__ttymode & ECHO)
X              {
X              if (*p == '\n' && (__ttymode & CRMOD))
X                   {
X                   console_write_byte(fd, '\n');
```

```
X                 console_write_byte(fd, '\r');
X                 __col_pos = 0;
X                 }
X                else
X                 (void) _echochar(*p);
X            }
X        if (++cnt >= nbytes || !(short)console_status(fd))
X            return cnt;
X        p++;
X        }
X       else
X   if (*p == __tchars[TC_LNEXTC])
X        {
X        if (__ttymode & ECHO)
X                {
X            console_write_byte(fd, '^');
X            console_write_byte(fd, '\b');
X            }
X        *p = console_read_byte(fd);
X        echochar(*p++);
X        cnt++;
X        }
X       else
X   if (*p == __tchars[TC_EOFC])
X        {
X        if (__ttymode & ECHO)
X            {
X            int i = _echochar(*p);
X            __col_pos -= i;
X            while (i-- > 0)
X                console_write_byte(fd, '\b');
X            }
X        return cnt;
X        }
X       else
X   if (*p == '\n' || *p == __tchars[TC_BRKC])
X        {
X        if (__ttymode & ECHO)
X            if (*p == '\n')
X                {
X                console_write_byte(fd, '\n');
X                if (__ttymode & CRMOD)
X                    {
X                    console_write_byte(fd, '\r');
X                    __col_pos = 0;
X                    }
X                }
X               else
X                (void) _echochar(*p);
X        return ++cnt;
X        }
X       else
X   if (*p == __tchars[TC_ERASE])
```

```
X        {
X        if (cnt)
X             {
X             p--;
X             delchar(--cnt);
X             }
X        }
X       else
X    if (*p == __tchars[TC_KILL])
X        {
X        while (--cnt >= 0)
X             {
X             delchar(cnt);
X             p--;
X             }
X        cnt = 0;
X        }
X       else
X    if (*p == __tchars[TC_WERASC])
X            {
X        p--;
X        while (cnt && iswhite(*p))
X                {
X             delchar(--cnt);
X             p--;
X                }
X        while (cnt && !iswhite(*p))
X                {
X             delchar(--cnt);
X             p--;
X                }
X        p++;
X            }
X       else
X    if (*p == __tchars[TC_RPRNTC])
X            {
X        unsigned char *s;
X
X        echochar(__tchars[TC_RPRNTC]);
X        console_write_byte(fd, '\r');
X        console_write_byte(fd, '\n');
X        __col_pos = 0;
X        start_col = 0;
X        if (__ttymode & ECHO)
X                for (s = buf ; s < p ; s++)
X                     echochar(*s);
X            }
X       else
X            {
X        echochar(*p++);
X        cnt++;
X        }
X        if (cnt >= nbytes)
```

```
X            return cnt;
X        }
X    /*NOTREACHED*/
X}
X
Xstatic  int
X_echochar(c, fd)
Xunsigned char c;
Xint fd;
X{
X  int   len = 0;
X
X  if (c & META_MASK)
X    {
X    console_write_byte(fd, 'M');
X    console_write_byte(fd, '-');
X    c &= ~META_MASK;
X    len += 2;
X    }
X  if (c < ' ')
X    {
X    if (c == '\t')
X         {
X         int  i;
X
X         len = ((__col_pos | 7) + 1) - __col_pos;
X         if (__ttymode & XTABS)
X              for (i = len ; i-- ;)
X                   console_write_byte(fd, ' ');
X             else
X              console_write_byte(fd, '\t');
X         }
X        else
X            {
X         console_write_byte(fd, '^');
X         console_write_byte(fd, c + 0x40);
X         len += 2;
X            }
X         }
X    else
X      if (c == DEL)
X    {
X    console_write_byte(fd, '^');
X    console_write_byte(fd, '?');
X    len += 2;
X    }
X      else
X    {
X    console_write_byte(fd, c);
X    len++;
X    }
X  __col_pos += len;
X  return len;
```

```
X}
X
Xstatic  void
X_delchar(n, fd)
Xint n;
Xint fd;
X{
X  int   len;
X  unsigned char   c = thebuf[n];
X
X  if (c & META_MASK) {
X    len = 2;
X    c &= ~META_MASK;
X  } else
X    len = 0;
X  if (c < ' ' || c == DEL)
X    {
X    if (c == '\t')
X         len = __col_pos - str_length(thebuf, n);
X        else
X         len += 2;
X    }
X    else
X    len++;
X
X  __col_pos -= len;
X  while (len--)
X    {
X#if USING_BCONIN
X    console_write_byte(fd, '\b');
X#endif
X    console_write_byte(fd, ' ');
X    console_write_byte(fd, '\b');
X    }
X}
X
Xstatic int str_length(p, n)
Xunsigned char     *p;
Xint n;
X{
X  int   pos = start_col;
X  unsigned char   c;
X
X  while (n--) {
X    c = *p++;
X    if (c & META_MASK)
X         {
X         pos += 2;
X         c &= ~META_MASK;
X         }
X        if (c < ' ' || c == DEL)
X         {
X         if (c == '\t')
```

```
X             pos = (pos | 7) + 1;
X            else
X             pos += 2;
X        }
X        else
X             pos++;
X    }
X  return pos;
X}
END_OF_FILE
if test 8951 -ne `wc -c <'read.c'`; then
    echo shar: \"'read.c'\" unpacked with wrong size!
fi
# end of 'read.c'
fi
echo shar: End of archive 5 \(of 7\).
cp /dev/null ark5isdone
MISSING=""
for I in 1 2 3 4 5 6 7 ; do
    if test ! -f ark${I}isdone ; then
     MISSING="${MISSING} ${I}"
    fi
done
if test "${MISSING}" = "" ; then
    echo You have unpacked all 7 archives.
    rm -f ark[1-9]isdone
else
    echo You still need to unpack the following archives:
    echo "        " ${MISSING}
fi
##  End of shell archive.
exit 0


From -5074557826077258777
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,37a122929869735d,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-08-19 10:53:54 PST
Path: utzoo!attcan!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!
cs.utexas.edu!sun-barr!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i077: gcclib2 -- Patches to GCC library  part06/07
Keywords: shar, C
Message-ID: <34139@grapevine.uucp>
Date: 19 Aug 89 17:53:54 GMT
Sender: news@grapevine.uucp
Lines: 924
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: uunet.UU.NET!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 2, Issue 77
```

Archive-name: gcclib2/part06

```
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of archive 6 (of 7)."
# Contents:  include/osbind.h
# Wrapped by roeder@sbsvax on Wed Aug 16 22:03:04 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'include/osbind.h' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'include/osbind.h'\"
else
echo shar: Extracting \"'include/osbind.h'\" \(20404 characters\)
sed "s/^X//" >'include/osbind.h' <<'END_OF_FILE'
X#ifndef _OSBIND_H
X#define _OSBIND_H
X/*
X *  OSBIND.H  Mnemonic names for operating system calls
X *
X * Adapted from Dale Schumacher's version to Mark Johnson C by Eric Gisin.
X */
X
X/* (er) 11.8.89 changed macros to inline functions */
X/*  this is ABSOLUTELY necessary for Super(),  */
X/*  because the changed stack would be changed */
X/*  back by the unlk instruction               */
X
X#pragma once
X
X/* ..sb correction of the calls 19/11/88 */
X/* hacked by jrd for GNU */
X
X/*
X *  GEMDOS   (trap1)
X */
X
Xstatic __inline volatile void
XPterm0(void)
X{
X    __asm("movew #0x00,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    /* we will never come back, so we don't have to correct the stack */
X}
X
Xstatic __inline long
XCconin(void)
X{
X    long ret;
```

```
X
X    __asm("movew #0x00,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void
XCconout(short c)
X{
X    __asm("movew %0,sp@-" :: "g" (c));
X    __asm("movew #0x02,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X}
X
Xstatic __inline long
XCauxin(void)
X{
X    long ret;
X
X    __asm("movew #0x03,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XCauxout(short c)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (c));
X    __asm("movew #0x04,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XCprnout(short c)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (c));
X    __asm("movew #0x05,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
```

```
X}
X
Xstatic __inline long
XCrawio(short data)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (data));
X    __asm("movew #0x06,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XCrawcin(void)
X{
X    long ret;
X
X    __asm("movew #0x07,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XCnecin(void)
X{
X    long ret;
X
X    __asm("movew #0x08,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void
XCconws(char *s)
X{
X    __asm("movel %0,sp@-" :: "g" (s));
X    __asm("movew #0x09,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X}
X
Xstatic __inline void
XCconrs(char *buf)
X{
X    __asm("movel %0,sp@-" :: "g" (buf));
X    __asm("movew #0x0A,sp@-");
```

```
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X}
X
Xstatic __inline short
XCconis(void)
X{
X    short    ret;
X
X    __asm("movew #0x0B,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline const void
XDsetdrv(short d)
X{
X    __asm("movew %0,sp@-" :: "g" (d));
X    __asm("movew #0x0E,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X}
X
Xstatic __inline short
XCconos(void)
X{
X    short    ret;
X
X    __asm("movew #0x10,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XCprnos(void)
X{
X    short    ret;
X
X    __asm("movew #0x11,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XCauxis(void)
X{
X    short    ret;
```

```
X
X    __asm("movew #0x12,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XCauxos(void)
X{
X    short    ret;
X
X    __asm("movew #0x13,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XDgetdrv(void)
X{
X    long ret;
X
X    __asm("movew #0x19,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XFsetdta(void *dta)
X{
X    long ret;
X
X    __asm("movel %0,sp@-" :: "g" (dta));
X    __asm("movew #0x1A,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
X/*
X * The next binding is not quite right if used in another than the usual ways:
X *  1. Super(1L) from either user or supervisor mode
X *  2. ret = Super(0L) from user mode and after this Super(ret) from
X *     supervisor mode
X * We get the following situations (usp, ssp relative to the start of Super):
X *  Parameter Userstack Superstack    Calling Mode  ret
X *     1L            usp           ssp            user   0L
```

```
X *     1L            usp           ssp         supervisor
     -1L
X *     0L           usp-6          usp            user ssp
X *     0L            ssp          ssp-6        supervisor   ssp-6
X *    ptr           usp-6         ptr+6           user ssp
X *    ptr           usp+6          ptr         supervisor
      sr
X * The usual C-bindings are safe only because the "unlk a6" is compensating
X * the errors when you invoke this function. In this binding the "unlk a6" at
X * the end of the calling function compensates the error made in sequence 2
X * above (the usp is 6 to low after the first call which is not corrected by
X * the second call).
X */
Xstatic __inline long
XSuper(long ptr)
X{
X    long ret;
X
X    __asm("movel %0,sp@-" :: "g" (ptr));
X    __asm("movew #0x20,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XTgetdate(void)
X{
X    long ret;
X
X    __asm("movew #0x2A,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void
XTsetdate(long date)
X{
X    __asm("movel %0,sp@-" :: "g" (date));
X    __asm("movew #0x2B,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X}
X
Xstatic __inline long
XTgettime(void)
X{
X    long ret;
X
X    __asm("movew #0x2C,sp@-");
```

```
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void
XTsettime(long time)
X{
X    __asm("movel %0,sp@-" :: "g" (time));
X    __asm("movew #0x2D,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X}
X
Xstatic __inline long
XFgetdta(void)
X{
X    long ret;
X
X    __asm("movew #0x2F,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline const short
XSversion(void)
X{
X    short    ret;
X
X    __asm("movew #0x30,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline volatile void
XPtermres(long save, short rv)
X{
X    __asm("movew %0,sp@-" :: "g" (rv));
X    __asm("movel %0,sp@-" :: "g" (save));
X    __asm("movew #0x31,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    /* if we don't call Ptermres(save,-32) and ignore SIGINT, */
X    /* we can't come back */
X    __asm("addql #8,sp");
X}
X
Xstatic __inline long
XDfree(void *buf, short d)
```

```
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (d));
X    __asm("movel %0,sp@-" :: "g" (buf));
X    __asm("movew #0x36,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #8,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XDcreate(char *path)
X{
X    short    ret;
X
X    __asm("movel %0,sp@-" :: "g" (path));
X    __asm("movew #0x39,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XDdelete(char *path)
X{
X    short    ret;
X
X    __asm("movel %0,sp@-" :: "g" (path));
X    __asm("movew #0x3A,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XDsetpath(char *path)
X{
X    short    ret;
X
X    __asm("movel %0,sp@-" :: "g" (path));
X    __asm("movew #0x3B,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XFcreate(char *fn,short mode)
```

```
X{
X    short    ret;
X
X    __asm("movew %0,sp@-" :: "g" (mode));
X    __asm("movel %0,sp@-" :: "g" (fn));
X    __asm("movew #0x3C,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #8,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XFopen(char *fn,short mode)
X{
X    short    ret;
X
X    __asm("movew %0,sp@-" :: "g" (mode));
X    __asm("movel %0,sp@-" :: "g" (fn));
X    __asm("movew #0x3D,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #8,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XFclose(short handle)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (handle));
X    __asm("movew #0x3E,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XFread(short handle,long cnt,char *buf)
X{
X    long ret;
X
X    __asm("movel %0,sp@-" :: "g" (buf));
X    __asm("movel %0,sp@-" :: "g" (cnt));
X    __asm("movew %0,sp@-" :: "g" (handle));
X    __asm("movew #0x3F,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #12,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
```

```
X
Xstatic __inline long
XFwrite(short handle,long cnt,char *buf)
X{
X    long ret;
X
X    __asm("movel %0,sp@-" :: "g" (buf));
X    __asm("movel %0,sp@-" :: "g" (cnt));
X    __asm("movew %0,sp@-" :: "g" (handle));
X    __asm("movew #0x40,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #12,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XFdelete(char *fn)
X{
X    short    ret;
X
X    __asm("movel %0,sp@-" :: "g" (fn));
X    __asm("movew #0x41,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XFseek(long where,short handle,short how)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (how));
X    __asm("movew %0,sp@-" :: "g" (handle));
X    __asm("movel %0,sp@-" :: "g" (where));
X    __asm("movew #0x42,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #10,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XFattrib(char *fn,short rwflag,short attr)
X{
X    short    ret;
X
X    __asm("movew %0,sp@-" :: "g" (attr));
X    __asm("movew %0,sp@-" :: "g" (rwflag));
X    __asm("movel %0,sp@-" :: "g" (fn));
X    __asm("movew #0x43,sp@-");
```

```
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #10,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XFdup(short handle)
X{
X    short    ret;
X
X    __asm("movew %0,sp@-" :: "g" (handle));
X    __asm("movew #0x45,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XFforce(short Hstd,short Hnew)
X{
X    short    ret;
X
X    __asm("movew %0,sp@-" :: "g" (Hnew));
X    __asm("movew %0,sp@-" :: "g" (Hstd));
X    __asm("movew #0x46,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XDgetpath(char *buf,short d)
X{
X    short    ret;
X
X    __asm("movew %0,sp@-" :: "g" (d));
X    __asm("movel %0,sp@-" :: "g" (buf));
X    __asm("movew #0x47,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #8,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XMalloc(long size)
X{
X    long ret;
X
X    __asm("movel %0,sp@-" :: "g" (size));
```

```
X    __asm("movew #0x48,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void
XMfree(void *ptr)
X{
X    __asm("movel %0,sp@-" :: "g" (ptr));
X    __asm("movew #0x49,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X}
X
Xstatic __inline short
XMshrink(void *ptr,long size)
X{
X    short    ret;
X
X    __asm("movel %0,sp@-" :: "g" (size));
X    __asm("movel %0,sp@-" :: "g" (ptr));
X    __asm("clrw sp@-");
X    __asm("movew #0x4A,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #12,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XPexec(short mode,char *prog,char *tail,char *env)
X{
X    long ret;
X
X    __asm("movel %0,sp@-" :: "g" (env));
X    __asm("movel %0,sp@-" :: "g" (tail));
X    __asm("movel %0,sp@-" :: "g" (prog));
X    __asm("movew %0,sp@-" :: "g" (mode));
X    __asm("movew #0x4B,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #16,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline /* volatile */ void
XPterm(short rv)
X{
X    __asm("movew %0,sp@-" :: "g" (rv));
X    __asm("movew #0x4C,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
```

```
X    /* if we don't call Pterm(-32) and ignore SIGINT, we can't come back */
X    __asm("addql #4,sp");
X}
X
Xstatic __inline short
XFsfirst(char *filespec,short attr)
X{
X    short    ret;
X
X    __asm("movew %0,sp@-" :: "g" (attr));
X    __asm("movel %0,sp@-" :: "g" (filespec));
X    __asm("movew #0x4E,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #8,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XFsnext(void)
X{
X    short    ret;
X
X    __asm("movew #0x4F,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XFrename(char *old,char *new)
X{
X    short    ret;
X
X    __asm("movel %0,sp@-" :: "g" (new));
X    __asm("movel %0,sp@-" :: "g" (old));
X    __asm("clrw sp@-");
X    __asm("movew #0x56,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #12,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void
XFdatime(long *timeptr,short handle,short rwflag)
X{
X    __asm("movew %0,sp@-" :: "g" (rwflag));
X    __asm("movew %0,sp@-" :: "g" (handle));
X    __asm("movel %0,sp@-" :: "g" (timeptr));
X    __asm("movew #0x57,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
```

```
X    __asm("addw #10,sp");
X}
X
X
X/* codes for Pexec */
X
X#define  PE_LOADGO     0         /* load & go */
X#define  PE_LOAD            1         /* just load */
X#define  PE_CBASEPAGE       2         /* just create basepage */
X#define  PE_GO              3         /* just go */
X
X/* zzz do the rest later */
X
X/*
X *  BIOS (trap13)
X */
X
Xstatic __inline void
XGetmpb(void *mpb)
X{
X    __asm("movew #0,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X}
X
Xstatic __inline short
XBconstat(short DEV)
X{
X    short    ret;
X
X    __asm("movew %0,sp@-" :: "g" (DEV));
X    __asm("movew #1,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XBconin(short DEV)
X{
X    short    ret;
X
X    __asm("movew %0,sp@-" :: "g" (DEV));
X    __asm("movew #2,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void
XBconout(short DEV,short c)
```

```
X{
X    __asm("movew %0,sp@-" :: "g" (c));
X    __asm("movew %0,sp@-" :: "g" (DEV));
X    __asm("movew #3,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X}
X
Xstatic __inline long
XRwabs(short rwflag,void *buf,short n,short sector,short d)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (d));
X    __asm("movew %0,sp@-" :: "g" (sector));
X    __asm("movew %0,sp@-" :: "g" (n));
X    __asm("movel %0,sp@-" :: "g" (buf));
X    __asm("movew %0,sp@-" :: "g" (rwflag));
X    __asm("movew #4,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #14,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void *
XSetexc(short vnum,void *vptr)
X{
X    void *ret;
X
X    __asm("movel %0,sp@-" :: "g" (vptr));
X    __asm("movew %0,sp@-" :: "g" (vnum));
X    __asm("movew #5,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #8,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XTickcal()
X{
X    long ret;
X
X    __asm("movew #6,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void *
XGetbpb(short d)
```

```
X{
X    void *ret;
X
X    __asm("movew %0,sp@-" :: "g" (d));
X    __asm("movew #7,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XBcostat(short DEV)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (DEV));
X    __asm("movew #8,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XMediach(short d)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (d));
X    __asm("movew #9,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XDrvmap(void)
X{
X    long ret;
X
X    __asm("movew #10,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XKbshift(short mode)
X{
X    long ret;
```

```
X
X    __asm("movew %0,sp@-" :: "g" (mode));
X    __asm("movew #11,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
X#define Getshift()    Kbshift(-1)
X
X/*
X * XBIOS    (trap14)
X */
X#define Initmous(type,param,vptr)   trap_14_wwll(0,type,param,vptr)
X#define Physbase()             trap_14_www(2)
X#define Logbase()          trap_14_www(3)
X#define Getrez()           (int)trap_14_www(4)
X#define Setscreen(lscrn,pscrn,rez)  trap_14_wllw(5,lscrn,pscrn,rez)
X#define Setpallete(palptr)     trap_14_wllw(6,palptr)
X#define Setcolor(colornum,mixture)  trap_14_wwll(7,colornum,mixture)
X#define Floprd(buf,x,d,sect,trk,side,n)
trap_14_wllwwwwwlw(8,buf,x,d,sect,trk,side,n)
X#define Flopwr(buf,x,d,sect,trk,side,n)
trap_14_wllwwwwwlw(9,buf,x,d,sect,trk,side,n)
X#define Flopfmt(b,x,d,spt,trk,sd,i,m,v)
trap_14_wllwwwwwlw(10,b,x,d,spt,trk,sd,i,m,v)
X#define Midiws(cnt,ptr)             trap_14_wwll(12,cnt,ptr)
X#define Mfpint(vnum,vptr)      trap_14_wwll(13,vnum,vptr)
X#define Iorec(ioDEV)           trap_14_wwll(14,ioDEV)
X#define Rsconf(baud,flow,uc,rs,ts,sc)
trap_14_wwwwwww(15,baud,flow,uc,rs,ts,sc)
X#define Keytbl(nrml,shft,caps)      trap_14_wlll(16,nrml,shft,caps)
X#define Random()           trap_14_www(17)
X#define Protobt(buf,serial,dsktyp,exec)
trap_14_wllwwwwwlw(18,buf,serial,dsktyp,exec)
X#define Flopver(buf,x,d,sect,trk,sd,n)
trap_14_wllwwwwwlw(19,buf,x,d,sect,trk,sd,n)
X#define Scrdmp()           trap_14_www(20)
X#define Cursconf(func,rate)         trap_14_www(21,func,rate)
X#define Settime(time)          trap_14_wllw(22,time)
X#define Gettime()          trap_14_www(23)
X#define Bioskeys()             trap_14_www(24)
X#define Ikbdws(len_minus1,ptr)      trap_14_wwll(25,len_minus1,ptr)
X#define Jdisint(vnum)          trap_14_www(26,vnum)
X#define Jenabint(vnum)             trap_14_www(27,vnum)
X#define Giaccess(data,reg)     trap_14_www(28,data,reg)
X#define Offgibit(ormask)       trap_14_www(29,ormask)
X#define Ongibit(andmask)       trap_14_www(30,andmask)
X#define Xbtimer(timer,ctrl,data,vnum)    trap_14_wwwwl(31,timer,ctrl,data,vnum)
X#define Dosound(ptr)           trap_14_wllw(32,ptr)
X#define Setprt(config)             trap_14_www(33,config)
X#define Kbdvbase()             trap_14_www(34)
```

```
X#define  Kbrate(delay,reprate)       trap_14_www(35,delay,reprate)
X#define  Prtblk(pblkptr)             trap_14_wllw(36,pblkptr)
X#define  Vsync()                 trap_14_www(37)
X#define  Supexec(funcptr)        trap_14_wllw(38,funcptr)
X#define Blitmode(flag)           trap_14_www(64,flag)
X#endif /* _OSBIND_H */
END_OF_FILE
if test 20404 -ne `wc -c <'include/osbind.h'`; then
    echo shar: \"'include/osbind.h'\" unpacked with wrong size!
fi
# end of 'include/osbind.h'
fi
echo shar: End of archive 6 \(of 7\).
cp /dev/null ark6isdone
MISSING=""
for I in 1 2 3 4 5 6 7 ; do
    if test ! -f ark${I}isdone ; then
     MISSING="${MISSING} ${I}"
    fi
done
if test "${MISSING}" = "" ; then
    echo You have unpacked all 7 archives.
    rm -f ark[1-9]isdone
else
    echo You still need to unpack the following archives:
    echo "        " ${MISSING}
fi
##  End of shell archive.
exit 0
```

```
From -2355018740160780433
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,2e6041a71cc73e3c,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-08-20 13:14:47 PST
Path: utzoo!attcan!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!
brutus.cs.uiuc.edu!apple!sun-barr!newstop!sun!grapevine!
koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i078: gcclib2 -- Patches to GCC library  part07/07
Keywords: shar, C
Message-ID: <34141@grapevine.uucp>
Date: 20 Aug 89 20:14:47 GMT
Sender: news@grapevine.uucp
Lines: 817
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: uunet.UU.NET!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 2, Issue 78
Archive-name: gcclib2/part07
```

```
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of archive 7 (of 7)."
# Contents:  malloc.c
# Wrapped by roeder@sbsvax on Wed Aug 16 22:03:05 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'malloc.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'malloc.c'\"
else
echo shar: Extracting \"'malloc.c'\" \(21197 characters\)
sed "s/^X//" >'malloc.c' <<'END_OF_FILE'
X/* Dynamic memory allocation for GNU.
X   Copyright (C) 1985 Richard M. Stallman,
X    based mostly on the public domain work of others.
X
XThis program is distributed in the hope that it will be useful,
Xbut without any warranty.  No author or distributor
Xaccepts responsibility to anyone for the consequences of using it
Xor for whether it serves any particular purpose or works at all,
Xunless he says so in writing.
X
X   Permission is granted to anyone to distribute verbatim copies
X   of this program's source code as received, in any medium, provided that
X   the copyright notice, the nonwarraty notice above
X   and this permission notice are preserved,
X   and that the distributor grants the recipient all rights
X   for further redistribution as permitted by this notice,
X   and informs him of these rights.
X
X   Permission is granted to distribute modified versions of this
X   program's source code, or of portions of it, under the above
X   conditions, plus the conditions that all changed files carry
X   prominent notices stating who last changed them and that the
X   derived material, including anything packaged together with it and
X   conceptually functioning as a modification of it rather than an
X   application of it, is in its entirety subject to a permission
X   notice identical to this one.
X
X   Permission is granted to distribute this program (verbatim or
X   as modified) in compiled or executable form, provided verbatim
X   redistribution is permitted as stated above for source code, and
X    A.  it is accompanied by the corresponding machine-readable
X      source code, under the above conditions, or
X    B.  it is accompanied by a written offer, with no time limit,
X      to distribute the corresponding machine-readable source code,
X      under the above conditions, to any one, in return for reimbursement
X      of the cost of distribution.   Verbatim redistribution of the
```

```
X      written offer must be permitted.  Or,
X    C.  it is distributed by someone who received only the
X      compiled or executable form, and is accompanied by a copy of the
X      written offer of source code which he received along with it.
X
X   Permission is granted to distribute this program (verbatim or as modified)
X   in executable form as part of a larger system provided that the source
X   code for this program, including any modifications used,
X   is also distributed or offered as stated in the preceding paragraph.
X
XIn other words, you are welcome to use, share and improve this program.
XYou are forbidden to forbid anyone else to use, share and improve
Xwhat you give them.   Help stamp out software-hoarding!  */
X
X
X/*
X * @(#)nmalloc.c 1 (Caltech) 2/21/82
X *
X *  U of M Modified: 20 Jun 1983 ACT: strange hacks for Emacs
X *
X *  Nov 1983, Mike@BRL, Added support for 4.1C/4.2 BSD.
X *
X * This is a very fast storage allocator.  It allocates blocks of a small
X * number of different sizes, and keeps free lists of each size.  Blocks
X * that don't exactly fit are passed up to the next larger size.  In this
X * implementation, the available sizes are (2^n)-4 (or -16) bytes long.
X * This is designed for use in a program that uses vast quantities of
X * memory, but bombs when it runs out.  To make it a little better, it
X * warns the user when he starts to get near the end.
X *
X * June 84, ACT: modified rcheck code to check the range given to malloc,
X * rather than the range determined by the 2-power used.
X *
X * Jan 85, RMS: calls malloc_warning to issue warning on nearly full.
X * No longer Emacs-specific; can serve as all-purpose malloc for GNU.
X * You should call malloc_init to reinitialize after loading dumped Emacs.
X * Call malloc_stats to get info on memory stats if MSTATS turned on.
X * realloc knows how to return same block given, just changing its size,
X * if the power of 2 is correct.
X */
X
X/*
X * nextf[i] is the pointer to the next free block of size 2^(i+3).  The
X * smallest allocatable block is 8 bytes.  The overhead information will
X * go in the first long of the block, and the returned pointer will point
X * to the second.
X *
X#ifdef MSTATS
X * nmalloc[i] is the difference between the number of mallocs and frees
X * for a given block size.
X#endif /* MSTATS */
X
X#define BUCKETS   43   /* this is sufficient for 16 MByte */
```

```
X
X#ifdef emacs
X#include "config.h"
X#endif /* emacs */
X
X#ifdef atarist
X#define atari
X#endif
X#ifndef atari
X#include <sys/param.h>
X#endif
X
X/* Determine which kind of system this is.  */
X#include <signal.h>
X#ifndef SIGTSTP
X#define USG
X#else /* SIGTSTP */
X#ifdef SIGIO
X#define BSD42
X#endif /* SIGIO */
X#endif /* SIGTSTP */
X
X#ifdef atari
X#include <osbind.h>
X#include <stdio.h>
X#define USG
X#define EXEC_PAGESIZE 256
X#define malloc_warning botch
X#define botch     Bconws
X#endif
X#ifdef __GNUC__
X#define gnu_malloc    malloc
X#define gnu_free  free
X#define gnu_realloc   realloc
X#define gnu_bcopy bcopy
X#define lsbrk         sbrk
X#ifdef atari
X#define sbrk      (char *) Malloc
X#endif
X#endif
X
X#ifndef BSD42
X#ifndef USG
X#include <sys/vlimit.h>         /* warn the user when near the end */
X#endif
X#else /* if BSD42 */
X#include <sys/time.h>
X#include <sys/resource.h>
X#endif /* BSD42 */
X
X
X#if defined (BSD4_1) || defined (USG)
X#ifdef EXEC_PAGESIZE
```

```
X#define getpagesize() EXEC_PAGESIZE
X#else
X#ifdef NBPG
X#define getpagesize() NBPG * CLSIZE
X#ifndef CLSIZE
X#define CLSIZE 1
X#endif /* no CLSIZE */
X#else /* no NBPG */
X#define getpagesize() NBPC
X#endif /* no NBPG */
X#endif /* no EXEC_PAGESIZE */
X#endif /* BSD4_1 or USG */
X
X#define ISALLOC ((unsigned char) 0xf7)   /* magic byte that implies allocation */
X#define ISFREE ((unsigned char) 0x54)    /* magic byte that implies free block */
X                       /* this is for error checking only */
X#define ISMEMALIGN ((unsigned char) 0xd6)     /* Stored before the value
X                               returned by memalign, with
X                               the rest of the word being
X                               the distance to the true
X                               beginning of the block.  */
X
Xextern char etext;
Xextern char *start_of_data ();
X
X/* These two are for user programs to look at, when they are interested.  */
X
Xlong malloc_sbrk_used;       /* amount of data space used now */
Xlong malloc_sbrk_unused;     /* amount more we can have */
X
X/* start of data space; can be changed by calling init_malloc */
Xstatic char *data_space_start;
X
X#ifdef MSTATS
Xstatic int nmalloc[BUCKETS];
Xstatic int nmaxalloc[BUCKETS];
Xstatic int nmal, nfre;
X#endif /* MSTATS */
X
X/* If range checking is not turned on, all we have is a flag indicating
X   whether memory is allocated, an index in nextf[], and a size field; to
X   realloc() memory we copy either size bytes or 1<<(index+3) bytes depending
X   on whether the former can hold the exact size (given the value of
X   'index').  If range checking is on, we always need to know how much space
X   is allocated, so the 'size' field is never used. */
X
Xstruct mhead {
X    unsigned char mh_alloc; /* ISALLOC or ISFREE */
X    unsigned char mh_index; /* index in nextf[] */
X/* Remainder are valid only when block is allocated */
X    unsigned short     mh_size;  /* size, if < 0x10000 */
X#ifdef rcheck
X    unsigned long mh_nbytes;    /* number of bytes allocated */
```

```
X    long     mh_magic4;    /* should be == MAGIC4 */
X#endif /* rcheck */
X};
X
X/* Access free-list pointer of a block.
X  It is stored at block + 4.
X  This is not a field in the mhead structure
X  because we want sizeof (struct mhead)
X  to describe the overhead for when the block is in use,
X  and we do not want the free-list pointer to count in that.  */
X
X#define CHAIN(a) \
X  (*(struct mhead **) (sizeof (char *) + (char *) (a)))
X
X#ifdef rcheck
X
X/* To implement range checking, we write magic values in at the beginning and
X   end of each allocated block, and make sure they are undisturbed whenever a
X   free or a realloc occurs. */
X/* Written in each of the 4 bytes following the block's real space */
X#define MAGIC1 ((unsigned char) 0x55)
X/* Written in the 4 bytes before the block's real space */
X#define MAGIC4 0x55555555L
X#ifdef __GNUC__
X#define ASSERT(p) if (!(p)) botch(#p "\n"); else
X#else
X#define ASSERT(p) if (!(p)) botch("p\n"); else
X#endif
X#define EXTRA  4      /* 4 bytes extra for MAGIC1s */
X#else
X#define ASSERT(p)
X#define EXTRA  0
X#endif /* rcheck */
X
X#define BUCKETSIZE(b) ((1L << ((b >> 1) + 2)) * (2 + b % 2))
X#define MIN_ALLOC 20   /* bucket with 8kB */
X
X/* nextf[i] is free list of blocks of size 2**(i + 3)  */
X
Xstatic struct mhead *nextf[BUCKETS];
X
X/* Number of bytes of writable memory we can expect to be able to get */
Xstatic long lim_data;
X/* Level number of warnings already issued.
X  0 -- no warnings issued.
X  1 -- 75% warning already issued.
X  2 -- 85% warning already issued.
X*/
Xstatic int warnlevel;
X
X/* nonzero once initial bunch of free blocks made */
Xstatic int   gotpool;
Xstatic long  poolsize;
```

```
X
X/* Cause reinitialization based on job parameters;
X  also declare where the end of pure storage is. */
Xmalloc_init (start)
X     char *start;
X{
X  register int    i;
X
X  data_space_start = start;
X  lim_data = 0L;
X  warnlevel = 0;
X  poolsize = 0;
X  for(i = 0; i < MIN_ALLOC; i++) poolsize += BUCKETSIZE(i);
X}
X

Xstatic  getpool();
X
Xstatic
Xmorecore (nu)               /* ask system for more memory */
X     register int nu;      /* size index to get more of  */
X{
X#ifndef __GNUC__
X  char *sbrk ();
X  char *lsbrk ();
X#endif
X  register char *cp;
X  register long siz;
X  register long block_size;
X  register long cur_size;
X
X  if (!data_space_start)
X    {
X#if defined(USG) && defined (emacs)
X      data_space_start = start_of_data ();
X#else /* not USG, or not Emacs */
X#ifdef atari
X#ifdef __GNUC__
X      malloc_init((char *) 8);  /* dummy */
X#else
X      data_space_start = sbrk(0);
X#endif
X#else
X      data_space_start = &etext;
X#endif atari
X#endif /* not USG, or not Emacs */
X    }
X
X#ifndef atari
X  if (lim_data == 0L)
X#endif
X    get_lim_data ();
X
```

```
X /* On initial startup, get two blocks of each size up to 1k bytes */
X  if (!gotpool) {
X    getpool (), getpool (), gotpool = 1;
X  }
X
X /* Find current end of memory and issue warning if getting near max */
X
X  malloc_sbrk_unused = lim_data - malloc_sbrk_used;
X
X  switch (warnlevel)
X    {
X    case 0:
X      if (malloc_sbrk_used > (lim_data / 4) * 3)
X    {
X      warnlevel++;
X      malloc_warning ("\nWarning: past 75% of memory limit\n");
X    }
X      break;
X    case 1:
X      if (malloc_sbrk_used > (lim_data / 20) * 17)
X    {
X      warnlevel++;
X      malloc_warning ("\nWarning: past 85% of memory limit\n");
X    }
X      break;
X    case 2:
X      if (malloc_sbrk_used > (lim_data / 20) * 19)
X    {
X      warnlevel++;
X      malloc_warning ("\nWarning: past 95% of memory limit\n");
X    }
X      break;
X    }
X
X  if(nextf[nu]) return;    /* perhaps we got what we wanted in getpool() */
X
X#ifdef ALLIGN_ON_K_BOUNDARY
X  cp = sbrk(0);
X  if ((int) cp & 0x3ff)    /* land on 1K boundaries */
X    sbrk (1024 - ((int) cp & 0x3ff));
X#endif /* ALLIGN_ON_K_BOUNDARY */
X
X /* Take at least 2k, and figure out how many blocks of the desired size
X    we're about to get */
X  for(siz = nu; siz <= MIN_ALLOC; siz++) {
X    if(nextf[siz]) {
X      cp = (char *) nextf[siz];
X      nextf[siz] = CHAIN(cp);
X      cur_size = BUCKETSIZE(siz);
X      goto insert_block;
X    }
X  }
X
```

```
X#ifdef __GNUC__
X  if (!(cp = (char *) Malloc (cur_size = BUCKETSIZE(siz))))
X#else
X  if ((cp = lsbrk (cur_size = BUCKETSIZE(siz))) == (char *) -1)
X#endif
X    return;           /* no more room! */
X  malloc_sbrk_used += BUCKETSIZE(siz);
X
X /* save new header and link the blocks together */
Xinsert_block:
X  block_size = BUCKETSIZE(nu);
X  while (siz >= 0)
X    {
X      CHAIN ((struct mhead *) cp) = (struct mhead *) nextf[nu];
X      nextf[nu] = (struct mhead *) cp;
X      ((struct mhead *) cp) -> mh_alloc = ISFREE;
X      ((struct mhead *) cp) -> mh_index = nu;
X      if((cur_size -= block_size) <= 0) break;
X      cp += block_size;
X      while((block_size = BUCKETSIZE(siz)) > cur_size) if(!(siz--)) break;
X      nu = siz;
X    }
X}
X
Xstatic
Xgetpool ()
X{
X  register int nu;
X  register char *cp;
X
X#ifdef ALLIGN_ON_K_BOUNDARY
X  cp = sbrk (0);
X  if ((int) cp & 0x3ff)    /* land on 1K boundaries */
X    sbrk (1024 - ((int) cp & 0x3ff));
X#endif /* ALLIGN_ON_K_BOUNDARY */
X
X  /* Get 2k of storage */
X
X  cp = sbrk (poolsize);
X#ifdef __GNUC__
X  if (!cp)
X#else
X  if (cp == (char *) -1)
X#endif
X    return;
X  malloc_sbrk_used += poolsize;
X
X  /* Divide it into an initial 8-word block
X     plus one block of size 2**nu for nu = 3 ... 10.  */
X
X#ifdef notdef
X  CHAIN (cp) = nextf[0];
X  nextf[0] = (struct mhead *) cp;
```

```
X  ((struct mhead *) cp) -> mh_alloc = ISFREE;
X  ((struct mhead *) cp) -> mh_index = 0;
X  cp += 8;
X#endif
X
X  for (nu = 0; nu < MIN_ALLOC; nu++)
X    {
X      CHAIN (cp) = nextf[nu];
X      nextf[nu] = (struct mhead *) cp;
X      ((struct mhead *) cp) -> mh_alloc = ISFREE;
X      ((struct mhead *) cp) -> mh_index = nu;
X      cp += BUCKETSIZE(nu);
X    }
X}
X

Xchar *
Xgnu_malloc (n)         /* get a block */
X     unsigned long n;
X{
X  register struct mhead *p;
X  register unsigned long nbytes;
X  register int nunits = 0;
X
X  /* Figure out how many bytes are required, rounding up to the nearest
X     multiple of 4, then figure out which nextf[] area to use */
X  nbytes = (n + sizeof *p + EXTRA + 3) & ~3L;
X  while(BUCKETSIZE(nunits) < nbytes) nunits++;
X
X  /* If there are no blocks of the appropriate size, go get some */
X  /* COULD SPLIT UP A LARGER BLOCK HERE ... ACT */
X  /* Below MIN_ALLOC morecore() splits up a larger block if there is one. */
X  /* If this can't be done, a block of BUCKETSIZE(MIN_ALLOC) is Malloc()'ed */
X  /* from the system. Above MIN_ALLOC we allocate the memory from the system */
X  /* and only if this fails we try to use a larger block (not splitting up). */
X  if (nextf[nunits] == 0L) {
X    morecore (nunits);
X    if (nextf[nunits] == 0L) {
X      /* try to find a bigger block to use */
X      register int    i;
X      for (i = nunits + 1; i < BUCKETS; i++) if(nextf[i]) break;
X      if(i < BUCKETS) nunits = i;
X    }
X  }
X
X  /* Get one block off the list, and set the new list head */
X  if ((p = nextf[nunits]) == 0L) {
X#ifdef MSTATS
X      printf("\nmalloc(%ld) [bucket #%d] failed with %ld bytes free (%d%% of %ld)
\n",
X        nbytes,nunits,Malloc(-1L),(int)(malloc_sbrk_unused * 100 / lim_data),
X        lim_data);
X      printf("nextf @ %06lx\n",nextf);
```

```
X      memory_statistic();
X      abort();
X#endif /* MSTATS */
X      return 0L;
X    }
X  nextf[nunits] = CHAIN (p);
X
X  /* Check for free block clobbered */
X  /* If not for this check, we would gobble a clobbered free chain ptr */
X  /* and bomb out on the NEXT allocate of this size block */
X  if (p -> mh_alloc != ISFREE || p -> mh_index != nunits)
X#ifdef rcheck
X    botch ("block on free list clobbered");
X#else /* not rcheck */
X    abort ();
X#endif /* not rcheck */
X
X  /* Fill in the info, and if range checking, set up the magic numbers */
X  p -> mh_alloc = ISALLOC;
X#ifdef rcheck
X  p -> mh_nbytes = n;
X  p -> mh_magic4 = MAGIC4;
X  {
X    register char  *m = (char *) (p + 1) + n;
X
X    *m++ = MAGIC1, *m++ = MAGIC1, *m++ = MAGIC1, *m = MAGIC1;
X  }
X#else /* not rcheck */
X  p -> mh_size = n;
X#endif /* not rcheck */
X#ifdef MSTATS
X  nmalloc[nunits]++;
X  if(nmaxalloc[nunits] < nmalloc[nunits]) nmaxalloc[nunits] = nmalloc[nunits];
X  nmal++;
X#endif /* MSTATS */
X  return (char *) (p + 1);
X}
X
Xgnu_free (mem)
X     char *mem;
X{
X  register struct mhead *p;
X  {
X    register char *ap = mem;
X
X    if (ap == 0L)
X      return;
X
X    p = (struct mhead *) ap;
X    p--;
X    if (p -> mh_alloc == ISMEMALIGN)
X      {
X    ap -= p->mh_size;
```

```
X    p = (struct mhead *) ap;
X    p--;
X      }
X
X    if (p -> mh_alloc != ISALLOC)
X      abort ();
X
X#ifdef rcheck
X    ASSERT (p -> mh_magic4 == MAGIC4);
X    ap += p -> mh_nbytes;
X    ASSERT (*ap++ == MAGIC1); ASSERT (*ap++ == MAGIC1);
X    ASSERT (*ap++ == MAGIC1); ASSERT (*ap   == MAGIC1);
X#endif /* rcheck */
X  }
X  {
X    register int nunits = p -> mh_index;
X
X    ASSERT (nunits < BUCKETS);
X    p -> mh_alloc = ISFREE;
X    CHAIN (p) = nextf[nunits];
X    nextf[nunits] = p;
X#ifdef MSTATS
X    nmalloc[nunits]--;
X    nfre++;
X#endif /* MSTATS */
X  }
X}
X
Xchar *
Xgnu_realloc (mem, n)
X     char *mem;
X     register unsigned long n;
X{
X  register struct mhead *p;
X  register unsigned long tocopy;
X  register long nbytes;
X  register int nunits;
X
X  if ((p = (struct mhead *) mem) == (struct mhead *) 0)
X    return gnu_malloc (n);
X  p--;
X  nunits = p -> mh_index;
X  ASSERT (p -> mh_alloc == ISALLOC);
X#ifdef rcheck
X  ASSERT (p -> mh_magic4 == MAGIC4);
X  {
X    register char *m = mem + (tocopy = p -> mh_nbytes);
X#ifdef TRACE
X    if(*m != MAGIC1) {
X      printf("m = %08lX\tp = %08lX\n",m,p);
X      printf("p->mh_alloc\t= %02x\n      index\t= %02x\n      size\t= %d\n
nbytes\t= %ld\n      magic\t= %08X\n",
X          (int) p->mh_alloc,(int) p->mh_index,p->mh_size,p->mh_nbytes,p-
```

```
>mh_magic4);
X      printf("m[0] = %02x\tm[1] = %02x\tm[2] = %02x\tm[3] = %02x\n",(int)m[0],
X        (int)m[1],(int)m[2],(int)m[3]);
X    }
X#endif /* TRACE */
X    ASSERT (*m++ == MAGIC1); ASSERT (*m++ == MAGIC1);
X    ASSERT (*m++ == MAGIC1); ASSERT (*m   == MAGIC1);
X  }
X#else /* not rcheck */
X  if (p -> mh_index >= 13)
X    tocopy = BUCKETSIZE(p -> mh_index) - sizeof *p;
X  else
X    tocopy = p -> mh_size;
X#endif /* not rcheck */
X
X  /* See if desired size rounds to same power of 2 as actual size. */
X  nbytes = (n + sizeof *p + EXTRA + 7) & ~7L;
X
X  /* If ok, use the same block, just marking its size as changed.  */
X  if (nbytes > BUCKETSIZE(nunits-1) && nbytes <= BUCKETSIZE(nunits))
X    {
X#ifdef rcheck
X      register char *m = mem + tocopy;
X      *m++ = 0;  *m++ = 0;  *m++ = 0;  *m++ = 0;
X      p-> mh_nbytes = n;
X      m = mem + n;
X      *m++ = MAGIC1;  *m++ = MAGIC1;  *m++ = MAGIC1;  *m++ = MAGIC1;
X#else /* not rcheck */
X      p -> mh_size = n;
X#endif /* not rcheck */
X      return mem;
X    }
X
X  if (n < tocopy)
X    tocopy = n;
X  {
X    register char *new;
X
X    if ((new = gnu_malloc (n)) == 0L) {
X#ifdef MSTATS
X      printf("realloc failing with %ld bytes of free memory\n",Malloc(-1L));
X      memory_statistic();
X#endif /* MSTATS */
X      return 0L;
X    }
X    gnu_bcopy (mem, new, tocopy);
X    gnu_free (mem);
X    return new;
X  }
X}
X
Xchar *
Xmemalign (alignment, size)
```

```
X     unsigned alignment;
X     unsigned long size;
X{
X  register char *ptr = gnu_malloc (size + alignment);
X  register char *aligned;
X  register struct mhead *p;
X
X  if (ptr == 0L)
X    return 0L;
X  /* If entire block has the desired alignment, just accept it.  */
X  if (((long) ptr & (alignment - 1)) == 0L)
X    return ptr;
X  /* Otherwise, get address of byte in the block that has that alignment.  */
X  aligned = (char *) (((long) ptr + alignment - 1) & -alignment);
X
X  /* Store a suitable indication of how to free the block,
X     so that free can find the true beginning of it.  */
X  p = (struct mhead *) aligned;
X  p--;
X  p -> mh_size = aligned - ptr;
X  p -> mh_alloc = ISMEMALIGN;
X  return aligned;
X}
X
X#ifndef HPUX
X/* This runs into trouble with getpagesize on HPUX.
X   Patching out seems cleaner than the ugly fix needed.  */
Xchar *
Xvalloc (size)
Xunsigned long    size;
X{
X  return memalign (getpagesize (), size);
X}
X#endif /* not HPUX */
X

X#ifdef MSTATS
X/* Return statistics describing allocation of blocks of size 2**n. */
X
Xstruct mstats_value
X  {
X    long blocksize;
X    int      nfree;
X    int      nused;
X    int      nmaxused;
X  };
X
Xstruct mstats_value
Xmalloc_stats (size)
X     int size;
X{
X  struct mstats_value v;
X  /* register int i; */
```

```
X  register struct mhead *p;
X
X  v.nfree = 0;
X
X  if (size < 0 || size >= BUCKETS)
X    {
X      v.blocksize = 0L;
X      if(size = -1) {
X    v.nused = v.nmaxused = nmal;
X    v.nfree = nfre;
X      } else v.nused = 0;
X      return v;
X    }
X
X  v.blocksize = BUCKETSIZE(size);
X  v.nused = nmalloc[size];
X  v.nmaxused = nmaxalloc[size];
X
X  for (p = nextf[size]; p; p = CHAIN (p))
X    if (p -> mh_alloc != ISFREE || p -> mh_index != size) {
X      v.nfree = -v.nfree;
X      break;
X    } else {
X      v.nfree++;
X    }
X
X  return v;
X}
X
Xmemory_statistic()
X{
X    int  i;
X    struct mstats_value    mv;
X
X    printf("statistics about allocated blocks of memory\n");
X    printf("used %6ld\tunused %ld\tfree %ld bytes",
X         malloc_sbrk_used,malloc_sbrk_unused,Malloc(-1L));
X    printf(" (%d%% of %ld)\n",(int)(malloc_sbrk_unused * 100 / lim_data),
X         lim_data);
X    mv = malloc_stats(-1);
X    printf("%d malloc\t%d free\n",mv.nused,mv.nfree);
X    printf("bucket\tblocksize\t# of used blocks (max)\t# of free blocks\n");
X    for(i = 0; i < BUCKETS; i++) {
X         mv = malloc_stats(i);
X         if(mv.nfree || mv.nused)
X              printf("%3d\t%7ld\t\t\t%4d (%d)\t\t%4d\n",
X                   i,mv.blocksize,mv.nused,mv.nmaxused,mv.nfree);
X    }
X}
X#endif /* MSTATS */
X

X/*
```

```
X * This function returns the total number of bytes that the process
X * will be allowed to allocate via the sbrk(2) system call.  On
X * BSD systems this is the total space allocatable to stack and
X * data.  On USG systems this is the data space only.
X */
X
X#ifdef atari
Xget_lim_data ()
X{
X    long ld;
X
X    if((ld = Malloc(-1L)) > malloc_sbrk_unused)
X         lim_data += ld - malloc_sbrk_unused;
X}
X
XBconws(str)
Xchar    *str;
X{
X    while(*str) {
X         if(*str == '\n') Bconout(2,'\r');
X         Bconout(2,*str++);
X    }
X}
X#else /* not Atari ST */
X#ifdef USG
X
Xget_lim_data ()
X{
X  extern long ulimit ();
X
X  lim_data = ulimit (3, 0);
X  lim_data -= (long) data_space_start;
X}
X
X#else /* not USG */
X#ifndef BSD42
X
Xget_lim_data ()
X{
X  lim_data = vlimit (LIM_DATA, -1);
X}
X
X#else /* BSD42 */
X
Xget_lim_data ()
X{
X  struct rlimit XXrlimit;
X
X  getrlimit (RLIMIT_DATA, &XXrlimit);
X  lim_data = XXrlimit.rlim_cur;     /* soft limit */
X}
X
X#endif /* BSD42 */
```

```
X#endif /* not USG */
X#endif /* Atari ST */
END_OF_FILE
if test 21197 -ne `wc -c <'malloc.c'`; then
    echo shar: \"'malloc.c'\" unpacked with wrong size!
fi
# end of 'malloc.c'
fi
echo shar: End of archive 7 \(of 7\).
cp /dev/null ark7isdone
MISSING=""
for I in 1 2 3 4 5 6 7 ; do
    if test ! -f ark${I}isdone ; then
     MISSING="${MISSING} ${I}"
    fi
done
if test "${MISSING}" = "" ; then
    echo You have unpacked all 7 archives.
    rm -f ark[1-9]isdone
else
    echo You still need to unpack the following archives:
    echo "        " ${MISSING}
fi
##  End of shell archive.
exit 0
```

```
From -6597852102123834043
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,f958d18e684036dc,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-08-20 15:12:55 PST
Path: utzoo!attcan!utgpu!jarvis.csri.toronto.edu!mailrus!ames!sun-barr!newstop!
sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i079: gcclib2-patch1 -- Patch for GCC library
Keywords: shar, diff, C
Message-ID: <34143@grapevine.uucp>
Date: 20 Aug 89 22:12:55 GMT
Sender: news@grapevine.uucp
Lines: 1589
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: uunet.UU.NET!unido!sbsvax!roeder (Edgar Roeder)
Posting-number: Volume 2, Issue 79
Archive-name: gcclib2-patch1

Once again Murphy has caught me. In the GCC library posted was a little bug.
When seeking on an output file (like stdout) the old code tried to read in a
new buffer from that 'file'. On stdout this means that on every fflush-call
the program reads one line of text. There was also a typo in the CR/LF
conversion of fgetc.
```

The files in the appended shar archive should replace the corresponding files
of the library.

     - Edgar

```
------------------------------------------------------------------------------
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      console.c
#      fgetc.c
#      fseek.c
#      read.c
#      include/osbind.h
#
sed 's/^X//' << 'SHAR_EOF' > console.c &&
X/* console IO stuff.  Abstracted out here cause I'm having so much trouble
X   getting Bconout and Cconout to work.
X
X   Well, after a lot of fooling around, it looks like I can almost get the
X   effect I want (single byte read with no echo) by doing Freads with
X   length 1.  That does everything right except that it echos printable
X   ascii chars; ie anything #x20 .. #x7F.  We kludge around that by
X   arranging for the caller (read) to know which chars have been echoed.
X   Sigh.
X
X   There seem to be some people in the atari st community who have never
X   heard about the st's character set. There are some chars > 0x7F which
X   can be printed. And they will be printed.
X   This code assumes that nowbody tries to read from stderr if stdin is
X   redirected to AUX:. (er)
X   If we use Crawcin()/Cconis() we avoid problems with typeahead.
X*/
X
X#include <osbind.h>
X#include <file.h>
X
Xint console_read_byte(gem_handle)
Xint gem_handle;
X{
X  /* char c[1]; */
X
X  return(Crawcin());       /* isatty(gem_handle) == TRUE */
X/*  Fread(gem_handle, 1L, &c[0]);
X  return(c[0] & 0xFF); */
X}
X
Xconsole_write_byte(gem_handle, c)
Xint gem_handle;
Xchar c;
X{
X  char ch[1];
```

```
X
X/*  if (gem_handle < 0)
X    Cconout(c);
X    else */
X    {
X    ch[0] = c;
X    Fwrite(gem_handle, 1L, &ch[0]);
X    }
X}
X
X/* return status (nbytes pending)
X   This is completely brain-dead.  always return 0 for now */
Xint console_status(gem_handle)
Xint gem_handle;
X{
X  return(Cconis());
X/*  return(0); */
X/*
X  if (gem_handle < 2)
X    return(Bconstat(2));
X    else
X  if (handle == 2)
X    return(Bconstat(1));
X    else
X    return(Bconstat(handle));   /* it'd sure be nice to know if this
X                          is right.  Too bad Atari doesn't
X                          want anyone to see their
X                          documentation... */
X}
SHAR_EOF
chmod 0644 console.c || echo "restore of console.c fails"
sed 's/^X//' << 'SHAR_EOF' > fgetc.c &&
X
X/* fgetc */
X
X#include <file.h>
X#include "std-guts.h"
X
Xstatic int fgetc_internal(f)
Xstruct file * f;
X{
X
X  if (!f) return(EOF);
X  if (f->eof_p) return(EOF);
X  if (!f->open_p) return(EOF);
X  if ((f->flags & 0x03) == O_WRONLY) return(EOF);
X  if (f->buf_max > f->buf_index)
X    {
X    int result = f->buf[f->buf_index] & 0xFF;
X
X    f->buf_index++;
X/*
X#ifdef DEBUG
```

```
X    fprintf(stderr, " >%02X", result);
X#endif
X*/
X    return(result);
X    }
X
X/* ok, get a new buf */
X  f->file_position += f->buf_max;    /* base pos for next buf */
X  f->buf_max = read(f->handle, &f->buf, BUFSIZE);
X  f->buf_index = 0;
X  if (f->buf_max <= f->buf_index)    /* didn't get anything? */
X    {
X    f->eof_p = 1;           /* remember the eof */
X    if (f->buf_max < 0)
X         f->last_file_error = f->buf_max;
X        else
X         f->last_file_error = 0;
X#ifdef DEBUG
X    fprintf(stderr, "fgetc(%X)->EOF\r\n", f);
X#endif
X    return(EOF);
X    }
X    else
X    {
X    int result = f->buf[f->buf_index] & 0xFF;
X
X    f->buf_index++;
X#ifdef DEBUG
X    fprintf(stderr, "fgetc(%X)->%02X\r\n", f, result);
X#endif
X    return(result);
X    }
X}
X
Xint fgetc(f)
Xstruct file * f;
X{
X  int ch;
X  int next_ch;
X
X  ch = fgetc_internal(f);
X  if ((ch == '\r') && !(f->flags & FILE_BINARY_P) && !isatty(fileno(f)))
X  /* ttys are doing this with CRMOD in read */
X    {
X    /* next char '\n'?  if so, just return the linefeed, else put it
X       back and return the cr */
X    next_ch = fgetc_internal(f);
X    if (next_ch == '\n')
X         ch = next_ch;
X        else
X         ungetc(next_ch, f);    /* doesn't deal with EOF right... BFD */
X    }
X  return(ch);
```

```
X}
SHAR_EOF
chmod 0644 fgetc.c || echo "restore of fgetc.c fails"
sed 's/^X//' << 'SHAR_EOF' > fseek.c &&
X#include <file.h>
X#include "std-guts.h"
X#include <errno.h>
X
Xlong fseek(f, pos, mode)
Xstruct file * f;
Xlong pos;
Xint mode;
X{
X  long new_pos;
X
X  f->eof_p = 0;                     /* just in case */
X  switch (mode)
X    {
X    case L_SET:  { new_pos = pos; break; };
X    case L_INCR: { new_pos = f->file_position + f->buf_index + pos;
X                mode = L_SET;
X                break; };
X    case L_XTND: { new_pos = pos; break; };
X    default:     { errno = EBADARG ; return(-1); };
X    };
X#ifdef DEBUG
X    fprintf(stderr, "fseek: %d,%d.  filepos %d, idx %d max %d\n",
X          pos, mode, f->file_position, f->buf_index, f->buf_max);
X#endif
X  if ((mode != L_XTND) &&       /* always really seek in extend mode */
X      (f->file_position <= new_pos) &&
X      ((f->file_position + f->buf_max) > new_pos))/* have it already? */
X    {
X    f->buf_index = new_pos - f->file_position;     /* yes, point at it */
X#ifdef DEBUG
X    fprintf(stderr, "  same buf: new idx %d\n", f->buf_index);
X#endif
X    return(0);
X    }
X  if (file_output_p(f))                      /* output; flush buf */
X    fflush(f);
X  f->buf_index = f->buf_max = 0;     /* flush buf */
X
X  if (file_input_p(f)) {
X/* the requested index is not in the current buffer.  if it's less than the
X   current file-position, assume we're walking backwards thru the file, and
X   set the real position to a half bufferful before the requested pos, and set
X   indices accordingly */
X
X  if (new_pos < f->file_position)
X    {
X    if (new_pos < (BUFSIZE / 2))     /* too close to a bufferful? */
X        {
```

```
X         f->buf_index = new_pos; /* say the idx is this far in */
X         new_pos = 0;       /*  and start at beginning */
X         }
X        else
X         {
X         new_pos -= ((BUFSIZE / 2) - 1);  /*  set up to seek to pos - bufsize + 1
*/
X         f->buf_index = (BUFSIZE / 2) - 1; /* and point at last byte */
X         }
X    }
X  new_pos = lseek(f->handle, new_pos, mode);
X  if (new_pos >= 0)
X    {
X    f->file_position = new_pos;
X    f->buf_max = read(f->handle, &f->buf, BUFSIZE);
X    }
X  } else {
X    new_pos = lseek(f->handle, new_pos, mode);
X    if (new_pos >= 0) f->file_position = new_pos;
X  }
X#ifdef DEBUG
X  fprintf(stderr, "  seeked to %d\n", f->file_position);
X#endif
X  if (new_pos < 0)
X    return (new_pos);
X    else
X    return (0);
X}
SHAR_EOF
chmod 0644 fseek.c || echo "restore of fseek.c fails"
sed 's/^X//' << 'SHAR_EOF' > read.c &&
X/*
X *      Cross Development System for Atari ST
X *     Copyright (c) 1988, Memorial University of Newfoundland
X *
X * $Header: read.c,v 1.2 88/02/03 22:56:30 m68k Exp $
X *
X * $Log: read.c,v $
X *
X * 1.5       er
X *
X * Added /dev/null dummy handle.
X * Fix it : When adding 'continue' after raise(SIGINT) and raise(SIGQUIT), get
X *      unexpected results in Emacs (perhaps problems with EOF in fgetc()).
X * Moved __col_pos to write.c (it's always included because of fflush in exit)
X *
X * 1.4       er
X *
X * Revision 1.4    89/01/15
X * Added support for function keys (they give an Esc-sequence with the
X * scan-code as parameter), META-key (Alternate sets the highest bit) and
X * O_NDELAY mode (no waiting if there is no available input). META-translation
X * is controled via the META-bit in __ttymode (set by ioctl()).
```

```
X *
X * 1.3       jrd
X *
X * Revision 1.2  88/02/03  22:56:30  m68k
X * Added unix like tty driver stuff
X *
X * Revision 1.1  88/01/29  17:31:39  m68k
X * Initial revision
X *
X */
X#include <osbind.h>
X#include <ioctl.h>
X#include <file.h>
X#include <errno.h>
X#include <signal.h>
X#include "tchars.h"
X
X#define META_MASK 0x80
X#define DEL       0x7f
X
X#define  iswhite(c)    ((c) == ' ' || (c) == '\t')
X#define  isvisable(c)  ((c) >= ' ' && (c) < DEL)
X#define  echochar(c)   if (__ttymode & ECHO) (void) _echochar(c, fd); else
X#define  delchar(n)    if (__ttymode & ECHO) _delchar(n, fd); else
X
X# ifndef NULL
X#  define    NULL 0
X# endif /* NULL */
X
Xstatic int str_length(unsigned char * p, int n);
X
Xextern int   __col_pos;
Xstatic int   start_col;
Xstatic unsigned char  *thebuf;
X
Xstatic  int  _echochar();
Xstatic  int  str_length();
Xstatic  void _delchar();
X
Xunion key {
X    int  i;
X    struct {
X         char shift_state;
X         char scan_code;
X         char filler;
X         unsigned char ascii_code;
X    } k;
X};
X
X/* possible shift_state's */
X#define  SHIFT_RIGHT   1
X#define  SHIFT_LEFT    2
X#define SHIFT     (SHIFT_LEFT|SHIFT_RIGHT)
```

```
X#define  CONTROL       4
X#define  ALTERNATE 8
X#define  CAPS_LOCK 16
X#define  MOUSE_RIGHT   32
X#define  MOUSE_LEFT    64
X#define  MOUSE     (MOUSE_LEFT|MOUSE_RIGHT)
X
Xstatic   enum { Esc, Number, Key }   _fn_state;
Xstatic   char _fn_key;
X
Xint
Xread(int fd, unsigned char *buf, int nbytes)
X{
X#ifdef DEBUG
X  char dbgbuf[64];
X#endif
X
X  int    rval;
X  int    cnt = 0;
X  unsigned char   *p = buf;
X  union key  ch;
X
X  if (fd == 1234 || nbytes <= 0) return(0);   /* /dev/null */
X  if ((fd >= 0) && (!isatty(fd)))
X    {
X    if ((rval = Fread(fd, nbytes, (char *) buf)) < 0)
X         {
X         errno = -rval;
X         rval = -1;
X         }
X#ifdef DEBUG
X    sprintf(dbgbuf, "read(%d, %X, %d)->%d\r\n", fd, buf, nbytes, rval);
X    dbgstr(dbgbuf);
X#endif
X    return rval;
X        }
X  if ((__handle_stat[fd] & O_NDELAY) && _fn_state == Esc && !console_status(fd))
{
X    errno = EAGAIN;
X    return -1;
X  }
X  thebuf = buf;
X  start_col = __col_pos;
X  while (1) {
X        switch (_fn_state) {
X                case Number :
X             *p = '#';
X             _fn_state = Key;
X             break;
X        case Key :
X             *p = _fn_key;
X             _fn_state = Esc;
X             break;
```

```
X        case Esc :
X             ch.i = console_read_byte(fd);
X             if((__ttymode & META) && !ch.k.shift_state) {
X                  struct sgttyb sg;
X
X                  ioctl(fd, TIOCGETP, &sg);
X                  ioctl(fd, TIOCSETN, &sg);
X                  ch.k.shift_state = Kbshift(-1);
X             }
X             if((__ttymode & META) &&
X                 (ch.k.ascii_code & META_MASK)) {
X                  /* these values are on the german keyboard */
X                  switch(ch.k.ascii_code) {
X                       case 0x81 :
X                            ch.k.ascii_code = '@'; break;
X                       case 0x84 :
X                            ch.k.ascii_code = ']'; break;
X                       case 0x8E :
X                            ch.k.ascii_code = '}'; break;
X                       case 0x94 :
X                            ch.k.ascii_code = '['; break;
X                       case 0x99 :
X                            ch.k.ascii_code = '{'; break;
X                       case 0x9A :
X                            ch.k.ascii_code = '\\'; break;
X                  }
X             }
X             if (ch.k.scan_code == 0x72 && ch.k.ascii_code == '\r')
X                  ch.k.ascii_code = '\n';
X             else
X             /* 3B = F1, 72 = Enter, 53 = Delete, 60 = '<' */
X             if ((ch.k.scan_code >= 0x3B && ch.k.scan_code <= 0x72
X               && ch.k.scan_code != 0x53 && ch.k.scan_code != 0x60))
X             {
X                  if (ch.k.ascii_code) _fn_key = ch.k.ascii_code;
X                  else _fn_key = ch.k.scan_code;
X                  ch.k.ascii_code = 0x1B; /* ESC */
X                  _fn_state = Number;
X             }
X             *p = ch.k.ascii_code;
X             /* Alternate as META-key sets the highest bit */
X             if ((__ttymode & META) &&
X                 (ch.k.shift_state & ALTERNATE)) {
X                  if (_fn_state != Esc) _fn_key |= META_MASK;
X                  else {
X                       if (ch.k.shift_state & SHIFT)
X             ch.k.ascii_code = __keytab->shift[ch.k.scan_code];
X                       else if (ch.k.shift_state & CAPS_LOCK)
X             ch.k.ascii_code = __keytab->capslock[ch.k.scan_code];
X                       else
X             ch.k.ascii_code = __keytab->unshift[ch.k.scan_code];
X                       if (!ch.k.ascii_code)
X             ch.k.ascii_code = ch.k.scan_code; /* Alt 0 - Alt 9 */
```

```
X                       if (ch.k.shift_state & CONTROL)
X                            ch.k.ascii_code &= 0x3F;
X                       *p = ch.k.ascii_code | META_MASK;
X                   }
X               }
X               break;
X           }
X       if (__ttymode & RAW)
X               {
X           if (__ttymode & ECHO)
X                   {
X               console_write_byte(fd, *p);
X               if (*p == '\r')
X                       __col_pos = 0;
X                   else
X               if (isvisable(*p))
X                    __col_pos++;
X               }
X           if (++cnt >= nbytes || !(short)console_status(fd))
X               return cnt;
X           p++;
X           continue;
X           }
X       if ((__ttymode & CRMOD) && *p == '\r')
X           *p = '\n';
X       if (*p == __tchars[TC_INTRC])
X           {
X           /* Do the bsd thing here, i.e. flush buffers
X            * and continue to read after the interupt
X            */
X           echochar(*p);
X           p = buf;
X           cnt = 0;
X           raise(SIGINT);
X           }
X          else
X       if (*p == __tchars[TC_QUITC])
X           {
X           echochar(*p);
X           p = buf;
X           cnt = 0;
X           raise(SIGQUIT);
X           }
X              else
X       if (*p == __tchars[TC_SUSPC])
X           {
X           echochar(*p);
X           p = buf;
X           cnt = 0;
X           raise(SIGTSTP);
X           continue;
X           }
X       if (__ttymode & CBREAK)
```

```
X        {
X        if (*p == __tchars[TC_LNEXTC])
X             *p = console_read_byte(fd);
X        if (__ttymode & ECHO)
X             {
X             if (*p == '\n' && (__ttymode & CRMOD))
X                  {
X                  console_write_byte(fd, '\n');
X                  console_write_byte(fd, '\r');
X                  __col_pos = 0;
X                  }
X                 else
X                  (void) _echochar(*p);
X             }
X        if (++cnt >= nbytes || !(short)console_status(fd))
X             return cnt;
X        p++;
X        }
X       else
X    if (*p == __tchars[TC_LNEXTC])
X        {
X        if (__ttymode & ECHO)
X                {
X             console_write_byte(fd, '^');
X             console_write_byte(fd, '\b');
X             }
X        *p = console_read_byte(fd);
X        echochar(*p++);
X        cnt++;
X        }
X       else
X    if (*p == __tchars[TC_EOFC])
X        {
X        if (__ttymode & ECHO)
X             {
X             int i = _echochar(*p);
X             __col_pos -= i;
X             while (i-- > 0)
X                  console_write_byte(fd, '\b');
X             }
X        return cnt;
X        }
X       else
X    if (*p == '\n' || *p == __tchars[TC_BRKC])
X        {
X        if (__ttymode & ECHO)
X             if (*p == '\n')
X                  {
X                  console_write_byte(fd, '\n');
X                  if (__ttymode & CRMOD)
X                       {
X                       console_write_byte(fd, '\r');
X                       __col_pos = 0;
```

```
X                       }
X                   }
X                  else
X                   (void) _echochar(*p);
X         return ++cnt;
X         }
X        else
X    if (*p == __tchars[TC_ERASE])
X         {
X         if (cnt)
X              {
X              p--;
X              delchar(--cnt);
X              }
X         }
X        else
X    if (*p == __tchars[TC_KILL])
X         {
X         while (--cnt >= 0)
X              {
X              delchar(cnt);
X              p--;
X              }
X         cnt = 0;
X         }
X        else
X    if (*p == __tchars[TC_WERASC])
X            {
X         p--;
X         while (cnt && iswhite(*p))
X                 {
X              delchar(--cnt);
X              p--;
X                 }
X         while (cnt && !iswhite(*p))
X                 {
X              delchar(--cnt);
X              p--;
X                 }
X         p++;
X            }
X        else
X    if (*p == __tchars[TC_RPRNTC])
X            {
X         unsigned char *s;
X
X         echochar(__tchars[TC_RPRNTC]);
X         console_write_byte(fd, '\r');
X         console_write_byte(fd, '\n');
X         __col_pos = 0;
X         start_col = 0;
X         if (__ttymode & ECHO)
X                 for (s = buf ; s < p ; s++)
```

```
X                       echochar(*s);
X                   }
X               else
X                   {
X                echochar(*p++);
X                cnt++;
X                }
X                if (cnt >= nbytes)
X                   return cnt;
X                }
X    /*NOTREACHED*/
X}
X
Xstatic  int
X_echochar(c, fd)
Xunsigned char c;
Xint fd;
X{
X  int   len = 0;
X
X  if (c & META_MASK)
X    {
X    console_write_byte(fd, 'M');
X    console_write_byte(fd, '-');
X    c &= ~META_MASK;
X    len += 2;
X    }
X  if (c < ' ')
X    {
X    if (c == '\t')
X         {
X         int  i;
X
X         len = ((__col_pos | 7) + 1) - __col_pos;
X         if (__ttymode & XTABS)
X              for (i = len ; i-- ;)
X                   console_write_byte(fd, ' ');
X              else
X               console_write_byte(fd, '\t');
X         }
X        else
X            {
X         console_write_byte(fd, '^');
X         console_write_byte(fd, c + 0x40);
X         len += 2;
X            }
X         }
X    else
X      if (c == DEL)
X    {
X    console_write_byte(fd, '^');
X    console_write_byte(fd, '?');
X    len += 2;
```

```
X    }
X      else
X    {
X    console_write_byte(fd, c);
X    len++;
X    }
X  __col_pos += len;
X  return len;
X}
X
Xstatic  void
X_delchar(n, fd)
Xint n;
Xint fd;
X{
X  int   len;
X  unsigned char   c = thebuf[n];
X
X  if (c & META_MASK) {
X    len = 2;
X    c &= ~META_MASK;
X  } else
X    len = 0;
X  if (c < ' ' || c == DEL)
X    {
X    if (c == '\t')
X         len = __col_pos - str_length(thebuf, n);
X        else
X         len += 2;
X    }
X    else
X    len++;
X
X  __col_pos -= len;
X  while (len--)
X    {
X    console_write_byte(fd, '\b');
X    console_write_byte(fd, ' ');
X    console_write_byte(fd, '\b');
X    }
X}
X
Xstatic int str_length(p, n)
Xunsigned char     *p;
Xint n;
X{
X  int   pos = start_col;
X  unsigned char   c;
X
X  while (n--) {
X    c = *p++;
X    if (c & META_MASK)
X         {
```

```
X         pos += 2;
X         c &= ~META_MASK;
X         }
X        if (c < ' ' || c == DEL)
X         {
X         if (c == '\t')
X              pos = (pos | 7) + 1;
X             else
X              pos += 2;
X         }
X         else
X              pos++;
X    }
X  return pos;
X}
SHAR_EOF
chmod 0644 read.c || echo "restore of read.c fails"
sed 's/^X//' << 'SHAR_EOF' > include/osbind.h &&
X#ifndef _OSBIND_H
X#define _OSBIND_H
X/*
X *  OSBIND.H  Mnemonic names for operating system calls
X *
X * Adapted from Dale Schumacher's version to Mark Johnson C by Eric Gisin.
X */
X
X/* (er) 11.8.89 changed macros to inline functions */
X/*  this is ABSOLUTELY necessary for Super(),  */
X/*  because the changed stack would be changed */
X/*  back by the unlk instruction               */
X
X#pragma once
X
X/* ..sb correction of the calls 19/11/88 */
X/* hacked by jrd for GNU */
X
X/*
X *  GEMDOS   (trap1)
X */
X
Xstatic __inline volatile void
XPterm0(void)
X{
X    __asm("movew #0x00,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    /* we will never come back, so we don't have to correct the stack */
X}
X
Xstatic __inline long
XCconin(void)
X{
X    long ret;
X
```

```
X    __asm("movew #0x00,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void
XCconout(short c)
X{
X    __asm("movew %0,sp@-" :: "g" (c));
X    __asm("movew #0x02,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X}
X
Xstatic __inline long
XCauxin(void)
X{
X    long ret;
X
X    __asm("movew #0x03,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void
XCauxout(short c)
X{
X    __asm("movew %0,sp@-" :: "g" (c));
X    __asm("movew #0x04,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X}
X
Xstatic __inline void
XCprnout(short c)
X{
X    __asm("movew %0,sp@-" :: "g" (c));
X    __asm("movew #0x05,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X}
X
Xstatic __inline long
XCrawio(short data)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (data));
X    __asm("movew #0x06,sp@-");
```

```
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XCrawcin(void)
X{
X    long ret;
X
X    __asm("movew #0x07,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XCnecin(void)
X{
X    long ret;
X
X    __asm("movew #0x08,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void
XCconws(char *s)
X{
X    __asm("movel %0,sp@-" :: "g" (s));
X    __asm("movew #0x09,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X}
X
Xstatic __inline void
XCconrs(char *buf)
X{
X    __asm("movel %0,sp@-" :: "g" (buf));
X    __asm("movew #0x0A,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X}
X
Xstatic __inline short
XCconis(void)
X{
X    short    ret;
X
```

```
X    __asm("movew #0x0B,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline const void
XDsetdrv(short d)
X{
X    __asm("movew %0,sp@-" :: "g" (d));
X    __asm("movew #0x0E,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X}
X
Xstatic __inline short
XCconos(void)
X{
X    short    ret;
X
X    __asm("movew #0x10,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XCprnos(void)
X{
X    short    ret;
X
X    __asm("movew #0x11,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XCauxis(void)
X{
X    short    ret;
X
X    __asm("movew #0x12,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
```

```
XCauxos(void)
X{
X    short    ret;
X
X    __asm("movew #0x13,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XDgetdrv(void)
X{
X    long ret;
X
X    __asm("movew #0x19,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XFsetdta(void *dta)
X{
X    long ret;
X
X    __asm("movel %0,sp@-" :: "g" (dta));
X    __asm("movew #0x1A,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
X/*
X * The next binding is not quite right if used in another than the usual ways:
X *  1. Super(1L) from either user or supervisor mode
X *  2. ret = Super(0L) from user mode and after this Super(ret) from
X *     supervisor mode
X * We get the following situations (usp, ssp relative to the start of Super):
X *  Parameter Userstack Superstack    Calling Mode  ret
X *     1L            usp           ssp            user  0L
X *     1L            usp           ssp         supervisor
     -1L
X *     0L           usp-6          usp            user ssp
X *     0L            ssp          ssp-6        supervisor   ssp-6
X *    ptr           usp-6         ptr+6           user ssp
X *    ptr           usp+6          ptr         supervisor
      sr
X * The usual C-bindings are safe only because the "unlk a6" is compensating
X * the errors when you invoke this function. In this binding the "unlk a6" at
```

```
X * the end of the calling function compensates the error made in sequence 2
X * above (the usp is 6 to low after the first call which is not corrected by
X * the second call).
X */
Xstatic __inline long
XSuper(long ptr)
X{
X    long ret;
X
X    __asm("movel %0,sp@-" :: "g" (ptr));
X    __asm("movew #0x20,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XTgetdate(void)
X{
X    long ret;
X
X    __asm("movew #0x2A,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void
XTsetdate(long date)
X{
X    __asm("movel %0,sp@-" :: "g" (date));
X    __asm("movew #0x2B,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X}
X
Xstatic __inline long
XTgettime(void)
X{
X    long ret;
X
X    __asm("movew #0x2C,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void
XTsettime(long time)
X{
```

```
X    __asm("movel %0,sp@-" :: "g" (time));
X    __asm("movew #0x2D,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X}
X
Xstatic __inline long
XFgetdta(void)
X{
X    long ret;
X
X    __asm("movew #0x2F,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline const short
XSversion(void)
X{
X    short    ret;
X
X    __asm("movew #0x30,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline volatile void
XPtermres(long save, short rv)
X{
X    __asm("movew %0,sp@-" :: "g" (rv));
X    __asm("movel %0,sp@-" :: "g" (save));
X    __asm("movew #0x31,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    /* if we don't call Ptermres(save,-32) and ignore SIGINT, */
X    /* we can't come back */
X    __asm("addql #8,sp");
X}
X
Xstatic __inline long
XDfree(void *buf, short d)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (d));
X    __asm("movel %0,sp@-" :: "g" (buf));
X    __asm("movew #0x36,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #8,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
```

```
X    return(ret);
X}
X
Xstatic __inline short
XDcreate(char *path)
X{
X    short    ret;
X
X    __asm("movel %0,sp@-" :: "g" (path));
X    __asm("movew #0x39,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XDdelete(char *path)
X{
X    short    ret;
X
X    __asm("movel %0,sp@-" :: "g" (path));
X    __asm("movew #0x3A,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XDsetpath(char *path)
X{
X    short    ret;
X
X    __asm("movel %0,sp@-" :: "g" (path));
X    __asm("movew #0x3B,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XFcreate(char *fn,short mode)
X{
X    short    ret;
X
X    __asm("movew %0,sp@-" :: "g" (mode));
X    __asm("movel %0,sp@-" :: "g" (fn));
X    __asm("movew #0x3C,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #8,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
```

```
X    return(ret);
X}
X
Xstatic __inline short
XFopen(char *fn,short mode)
X{
X    short    ret;
X
X    __asm("movew %0,sp@-" :: "g" (mode));
X    __asm("movel %0,sp@-" :: "g" (fn));
X    __asm("movew #0x3D,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #8,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XFclose(short handle)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (handle));
X    __asm("movew #0x3E,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XFread(short handle,long cnt,char *buf)
X{
X    long ret;
X
X    __asm("movel %0,sp@-" :: "g" (buf));
X    __asm("movel %0,sp@-" :: "g" (cnt));
X    __asm("movew %0,sp@-" :: "g" (handle));
X    __asm("movew #0x3F,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #12,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XFwrite(short handle,long cnt,char *buf)
X{
X    long ret;
X
X    __asm("movel %0,sp@-" :: "g" (buf));
X    __asm("movel %0,sp@-" :: "g" (cnt));
X    __asm("movew %0,sp@-" :: "g" (handle));
```

```
X    __asm("movew #0x40,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #12,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XFdelete(char *fn)
X{
X    short    ret;
X
X    __asm("movel %0,sp@-" :: "g" (fn));
X    __asm("movew #0x41,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XFseek(long where,short handle,short how)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (how));
X    __asm("movew %0,sp@-" :: "g" (handle));
X    __asm("movel %0,sp@-" :: "g" (where));
X    __asm("movew #0x42,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #10,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XFattrib(char *fn,short rwflag,short attr)
X{
X    short    ret;
X
X    __asm("movew %0,sp@-" :: "g" (attr));
X    __asm("movew %0,sp@-" :: "g" (rwflag));
X    __asm("movel %0,sp@-" :: "g" (fn));
X    __asm("movew #0x43,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #10,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XFdup(short handle)
X{
```

```
X    short    ret;
X
X    __asm("movew %0,sp@-" :: "g" (handle));
X    __asm("movew #0x45,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XFforce(short Hstd,short Hnew)
X{
X    short    ret;
X
X    __asm("movew %0,sp@-" :: "g" (Hnew));
X    __asm("movew %0,sp@-" :: "g" (Hstd));
X    __asm("movew #0x46,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XDgetpath(char *buf,short d)
X{
X    short    ret;
X
X    __asm("movew %0,sp@-" :: "g" (d));
X    __asm("movel %0,sp@-" :: "g" (buf));
X    __asm("movew #0x47,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #8,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XMalloc(long size)
X{
X    long ret;
X
X    __asm("movel %0,sp@-" :: "g" (size));
X    __asm("movew #0x48,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void
XMfree(void *ptr)
```

```
X{
X    __asm("movel %0,sp@-" :: "g" (ptr));
X    __asm("movew #0x49,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X}
X
Xstatic __inline short
XMshrink(void *ptr,long size)
X{
X    short    ret;
X
X    __asm("movel %0,sp@-" :: "g" (size));
X    __asm("movel %0,sp@-" :: "g" (ptr));
X    __asm("clrw sp@-");
X    __asm("movew #0x4A,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #12,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XPexec(short mode,char *prog,char *tail,char *env)
X{
X    long ret;
X
X    __asm("movel %0,sp@-" :: "g" (env));
X    __asm("movel %0,sp@-" :: "g" (tail));
X    __asm("movel %0,sp@-" :: "g" (prog));
X    __asm("movew %0,sp@-" :: "g" (mode));
X    __asm("movew #0x4B,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #16,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline /* volatile */ void
XPterm(short rv)
X{
X    __asm("movew %0,sp@-" :: "g" (rv));
X    __asm("movew #0x4C,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    /* if we don't call Pterm(-32) and ignore SIGINT, we can't come back */
X    __asm("addql #4,sp");
X}
X
Xstatic __inline short
XFsfirst(char *filespec,short attr)
X{
X    short    ret;
X
```

```
X    __asm("movew %0,sp@-" :: "g" (attr));
X    __asm("movel %0,sp@-" :: "g" (filespec));
X    __asm("movew #0x4E,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #8,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XFsnext(void)
X{
X    short    ret;
X
X    __asm("movew #0x4F,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline short
XFrename(char *old,char *new)
X{
X    short    ret;
X
X    __asm("movel %0,sp@-" :: "g" (new));
X    __asm("movel %0,sp@-" :: "g" (old));
X    __asm("clrw sp@-");
X    __asm("movew #0x56,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #12,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void
XFdatime(long *timeptr,short handle,short rwflag)
X{
X    __asm("movew %0,sp@-" :: "g" (rwflag));
X    __asm("movew %0,sp@-" :: "g" (handle));
X    __asm("movel %0,sp@-" :: "g" (timeptr));
X    __asm("movew #0x57,sp@-");
X    __asm("trap #1" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #10,sp");
X}
X
X
X/* codes for Pexec */
X
X#define  PE_LOADGO     0        /* load & go */
X#define  PE_LOAD           1        /* just load */
X#define  PE_CBASEPAGE      2        /* just create basepage */
```

```
X#define  PE_GO              3         /* just go */
X
X/* zzz do the rest later */
X
X/*
X *  BIOS (trap13)
X */
X
Xstatic __inline void
XGetmpb(void *mpb)
X{
X    __asm("movew #0,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X}
X
Xstatic __inline short
XBconstat(short DEV)
X{
X    short    ret;
X
X    __asm("movew %0,sp@-" :: "g" (DEV));
X    __asm("movew #1,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movew d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XBconin(short DEV)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (DEV));
X    __asm("movew #2,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void
XBconout(short DEV,short c)
X{
X    __asm("movew %0,sp@-" :: "g" (c));
X    __asm("movew %0,sp@-" :: "g" (DEV));
X    __asm("movew #3,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #6,sp");
X}
X
Xstatic __inline long
```

```
XRwabs(short rwflag,void *buf,short n,short sector,short d)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (d));
X    __asm("movew %0,sp@-" :: "g" (sector));
X    __asm("movew %0,sp@-" :: "g" (n));
X    __asm("movel %0,sp@-" :: "g" (buf));
X    __asm("movew %0,sp@-" :: "g" (rwflag));
X    __asm("movew #4,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addw #14,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void *
XSetexc(short vnum,void *vptr)
X{
X    void *ret;
X
X    __asm("movel %0,sp@-" :: "g" (vptr));
X    __asm("movew %0,sp@-" :: "g" (vnum));
X    __asm("movew #5,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #8,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XTickcal(void)
X{
X    long ret;
X
X    __asm("movew #6,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline void *
XGetbpb(short d)
X{
X    void *ret;
X
X    __asm("movew %0,sp@-" :: "g" (d));
X    __asm("movew #7,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
```

```
X}
X
Xstatic __inline long
XBcostat(short DEV)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (DEV));
X    __asm("movew #8,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XMediach(short d)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (d));
X    __asm("movew #9,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XDrvmap(void)
X{
X    long ret;
X
X    __asm("movew #10,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #2,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
Xstatic __inline long
XKbshift(short mode)
X{
X    long ret;
X
X    __asm("movew %0,sp@-" :: "g" (mode));
X    __asm("movew #11,sp@-");
X    __asm("trap #13" ::: "d0", "d1", "d2", "a0", "a1", "a2");
X    __asm("addql #4,sp");
X    __asm volatile ("movel d0,%0" : "=g" (ret));
X    return(ret);
X}
X
```

```
X#define Getshift()    Kbshift(-1)
X
X/*
X * XBIOS    (trap14)
X */
X#define Initmous(type,param,vptr)   trap_14_wwll(0,type,param,vptr)
X#define Physbase()              trap_14_www(2)
X#define Logbase()          trap_14_www(3)
X#define Getrez()           (int)trap_14_www(4)
X#define Setscreen(lscrn,pscrn,rez)  trap_14_wllw(5,lscrn,pscrn,rez)
X#define Setpallete(palptr)     trap_14_wllw(6,palptr)
X#define Setcolor(colornum,mixture)  trap_14_wwll(7,colornum,mixture)
X#define Floprd(buf,x,d,sect,trk,side,n)
trap_14_wllwwwwwlw(8,buf,x,d,sect,trk,side,n)
X#define Flopwr(buf,x,d,sect,trk,side,n)
trap_14_wllwwwwwlw(9,buf,x,d,sect,trk,side,n)
X#define Flopfmt(b,x,d,spt,trk,sd,i,m,v)
trap_14_wllwwwwwlw(10,b,x,d,spt,trk,sd,i,m,v)
X#define Midiws(cnt,ptr)             trap_14_wwll(12,cnt,ptr)
X#define Mfpint(vnum,vptr)      trap_14_wwll(13,vnum,vptr)
X#define Iorec(ioDEV)           trap_14_wwll(14,ioDEV)
X#define Rsconf(baud,flow,uc,rs,ts,sc)
trap_14_wwwwwww(15,baud,flow,uc,rs,ts,sc)
X#define Keytbl(nrml,shft,caps)      trap_14_wlll(16,nrml,shft,caps)
X#define Random()           trap_14_www(17)
X#define Protobt(buf,serial,dsktyp,exec)
trap_14_wllwwwwwlw(18,buf,serial,dsktyp,exec)
X#define Flopver(buf,x,d,sect,trk,sd,n)
trap_14_wllwwwwwlw(19,buf,x,d,sect,trk,sd,n)
X#define Scrdmp()           trap_14_www(20)
X#define Cursconf(func,rate)         trap_14_www(21,func,rate)
X#define Settime(time)          trap_14_wllw(22,time)
X#define Gettime()          trap_14_www(23)
X#define Bioskeys()             trap_14_www(24)
X#define Ikbdws(len_minus1,ptr)      trap_14_wwll(25,len_minus1,ptr)
X#define Jdisint(vnum)          trap_14_www(26,vnum)
X#define Jenabint(vnum)             trap_14_www(27,vnum)
X#define Giaccess(data,reg)     trap_14_www(28,data,reg)
X#define Offgibit(ormask)       trap_14_www(29,ormask)
X#define Ongibit(andmask)       trap_14_www(30,andmask)
X#define Xbtimer(timer,ctrl,data,vnum)    trap_14_wwwwl(31,timer,ctrl,data,vnum)
X#define Dosound(ptr)           trap_14_wllw(32,ptr)
X#define Setprt(config)             trap_14_www(33,config)
X#define Kbdvbase()             trap_14_www(34)
X#define Kbrate(delay,reprate)      trap_14_www(35,delay,reprate)
X#define Prtblk(pblkptr)            trap_14_wllw(36,pblkptr)
X#define Vsync()                trap_14_www(37)
X#define Supexec(funcptr)       trap_14_wllw(38,funcptr)
X#define Blitmode(flag)         trap_14_www(64,flag)
X#endif /* _OSBIND_H */
SHAR_EOF
chmod 0644 include/osbind.h || echo "restore of include/osbind.h fails"
exit 0
```

From -5441274389998250474
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,d67494f433d9da89,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-08-20 15:13:46 PST
Path: utzoo!attcan!utgpu!jarvis.csri.toronto.edu!mailrus!ames!sun-barr!newstop!
sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i080: mandel -- Fast Mandelbrot set generator
Keywords: shar, 68000
Message-ID: <34144@grapevine.uucp>
Date: 20 Aug 89 22:13:46 GMT
Sender: news@grapevine.uucp
Lines: 1257
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: SQ79%liverpool.ac.uk@NSFnet-Relay.AC.UK (Mark Powell)
Posting-number: Volume 2, Issue 80
Archive-name: mandel

Here is the source to the Mandelbrot generator that has just been posted
to the binaries newsgroup. Sorry about the lack of comments, but thats
just the sloppy way I program.

Mark Powell

ARPAnet : sq79%liv.ac.uk@{ucl-cs.arpa,cs.ucl.ac.uk}
USENET : ...!mcvax!ukc!liv.ac.uk!sq79
----------------------cut here---
```
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of shell archive."
# Contents:  MANDEL.S
# Wrapped by koreth@panarthea on Sun Aug 20 15:06:22 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'MANDEL.S' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'MANDEL.S'\"
else
echo shar: Extracting \"'MANDEL.S'\" \(17097 characters\)
sed "s/^X//" >'MANDEL.S' <<'END_OF_FILE'
X
Xtemp    equ  $30
X
X    move.l   4(sp),a5
X    move.l   #end+4096,d0
X    move.l   a5,d1
```

```
X    add.l    d0,d1
X    bclr #0,d1
X    move.l   d1,sp
X    move.l   d0,-(sp)
X    move.l   a5,-(sp)
X    pea  $4a0000
X    trap #1
X    lea  12(sp),sp
X    tst  d0
X    beq.s    okay
X
X    lea  nomem(pc),a1
X    bsr  prmes
X    bsr  anykey
X    bra  exit
X
Xokay    clr.l    -(sp)
X    move #32,-(sp)
X    trap #1
X    addq.l   #6,sp
X    move.l   d0,oldssp
X
X    dc.w $a00a
X
X    movem.l  $8240.w,d0-d7
X    movem.l  d0-d7,oldpal
X
X    pea  -1.w
X    move.l   (sp),-(sp)
X    move.l   (sp),-(sp)
X    move #16,-(sp)
X    trap #14
X    lea  14(sp),sp
X    move.l   d0,a0
X    move.l   8(a0),keytable
X
X    move #2,-(sp)
X    trap #14
X    addq.l   #2,sp
X    move.l   d0,phybase
X    move #3,-(sp)
X    trap #14
X    addq.l   #2,sp
X    move.l   d0,logbase
X
X    move #4,-(sp)
X    trap #14
X    addq.l   #2,sp
X    move d0,oldres
X    cmp  #2,d0
X    bne.s    colour
X
X    lea  nomono(pc),a1
```

```
X    bsr  prmes
X    bsr  anykey
X    bra  exit1
X
Xcolour   clr  -(sp)
X    pea  -1.w
X    move.l   (sp),-(sp)
X    move #5,-(sp)
X    trap #14
X    lea  12(sp),sp
X
X    move #$2700,sr
X    move.l   $fa06.w,oldier
X    clr.b    $fa07.w
X    move.b   #$40,$fa09.w
X    bclr #6,$fa0d.w
X    bclr #6,$fa11.w
X    bset #6,$fa15.w
X    move.l   $118.w,oldkeyb
X    move.l   #keybrd,$118.w
X    move.l   $456.w,a0
X    move.l   (a0),oldvblq
X    move.l   #vbl,(a0)
X    move.l   #palette,$45a.w
X    move #$2300,sr
X
X    lea  keydat(pc),a0
X    moveq    #3-1,d0
X    bsr  sendikbd
X
Xdrawfull
X    move.l   #$a0000000,a3
X    move.l   #$c0000000,a4
X    move.l   #$a3d70a,a1
X
X    bsr  mandel
X
X    bsr  mouseon
X
X* Main loop
X
Xmain    bsr  wvbl
X
X    move keys(pc),d0
X
X    cmp  #$13,d0       'r' redraw current
X    beq  domand
X    cmp  #1,d0         esc exit from program
X    beq  desktop
X    cmp  #$17,d0       'i' alter iterations
X    bne  noti
X
X* Alter iteration
```

```
X
X    movem.l  a1/a3/a4,-(sp)
X    bsr  screendown
X    lea  inpmes(pc),a1
X    bsr  prmes
X
Xwni cmp  #$17,keys
X    beq.s    wni
X
X    lea  inpbuf(pc),a5
X    move.l   a5,a6
Xnokey    move keys(pc),d7
X    cmp  #$74,d7
X    bge.s    nokey
X    move.l   keytable(pc),a0
X    move.b   0(a0,d7),d1
X    beq.s    nokey
X    cmp.b    #13,d1
X    beq.s    return
X    cmp.b    #8,d1
X    beq.s    delete
X    cmp.l    #inpbuf+10,a6
X    beq.s    nokey
X    cmp.b    #'0',d1
X    blt.s    nokey
X    cmp.b    #'9',d1
X    bgt.s    nokey
X    move.b   d1,(a6)+
X    move d1,d0
X    bsr  prchar
Xwaitnk   cmp  keys(pc),d7
X    beq.s    waitnk
X    bra.s    nokey
X
Xdelete   cmp.l    a5,a6
X    beq.s    nokey
X    subq.l   #1,a6
X    lea  delmes(pc),a1
X    bsr  prmes
X    bra.s    waitnk
X
Xreturn   cmp.l    a5,a6         anything in the buffer?
X    beq.s    restscr       if not then leave iterations unchanged
X
X    moveq    #0,d0
X    moveq    #0,d1
Xnxtchar  move.b   (a5)+,d0
X    sub  #'0',d0
X    move.l   d1,d2
X    add.l    d1,d1
X    add.l    d1,d1
X    add.l    d2,d1
X    add.l    d1,d1
```

```
X    add.l    d0,d1
X    cmp.l    a5,a6
X    bne.s    nxtchar
X    move.l   d1,iterations+2
X
Xrestscr moveq    #27,d0        put screen back up
X    bsr  prchar
X    moveq    #'f',d0
X    bsr  prchar
X    bsr  screenup
X    movem.l  (sp)+,a1/a3/a4
X
Xnoti    cmp  #$21,d0       'f' draw full set
X    bne.s    notf
X
X    bsr  mouseoff
X    bra  drawfull
X
Xnotf    cmp  #$61,d0       'undo' redraw last picture
X    bne.s    notundo
X
X    movem.l  lastmand(pc),a1/a3/a4
X    bra  domand
X
Xnotundo cmp  #$2d,d0
X    bne.s    notx
X
X    move.l   a1,-(sp)
X    bsr  screendown
X    lea  xmes(pc),a1
X    bsr  prmes
X    move.l   a3,d0
X    bsr  prhex32
X    bsr  space
X    move.l   a4,d0
X    bsr  prhex32
X    bsr  space
X    move.l   (sp),d0
X    bsr  prhex32
X    bsr  space
Xwaitnx  cmp  #$2d,keys
X    beq.s    waitnx
Xwaitkey tst  keys
X    beq.s    waitkey
X    bsr  screenup
X    move.l   (sp)+,a1
X
Xnotx    cmp  #$39,d0       space
X    bne  main
X
X    moveq    #0,d4
X    move box_width(pc),d4
X    move.l   a1,d0
```

```
X    move d0,d1
X    mulu d4,d1
X    swap d0
X    mulu d4,d0
X    swap d0
X    clr  d0
X    add.l    d0,d1
X
X* divide d1 by 200
X    move.l   #200,d5
X    moveq    #0,d2
X    moveq    #32-2,d3
X    addq #1,d3
Xdivlp   addx.l   d1,d1
X    addx.l   d2,d2
X    sub.l    d5,d2
X    bcc.s    subok
X    add.l    d5,d2
Xsubok   eor  #$10,sr
X    dbra d3,divlp
X    addx.l   d1,d1
X    tst.l    d1
X    bne.s    zoomok
X
X    move #$2700,sr
X    not  $8240.w
X    lea  $8800.w,a0
X    lea  beepdat(pc),a2
X    moveq    #7-1,d0
Xsndlp   move (a2)+,d1
X    movep    d1,0(a0)
X    dbra d0,sndlp
X    not  $8240.w
X    move #$2300,sr
X    bra  main
X
Xzoomok  movem.l  a1/a3/a4,lastmand
X
X    move mousex(pc),d2 Calculate top left corner of box, position
X    move mousey(pc),d3 in d2,d3
X    sub  d4,d3
X    lsl  #3,d4
X    divu #5,d4
X    sub  d4,d2
X
X    move.l   a1,d4
X    move d4,d5
X    mulu d2,d5
X    swap d4
X    mulu d2,d4
X    swap d4
X    clr  d4
X    add.l    d4,d5
```

```
X    add.l    d5,a3    a3=a3+x*a1
X
X    move.l   a1,d4
X    move d4,d5
X    mulu d3,d5
X    swap d4
X    mulu d3,d4
X    swap d4
X    clr  d4
X    add.l    d4,d5
X    add.l    d5,a4    a4=a4+y*a1
X
X    move.l   d1,a1
X
Xdomand  bsr  mouseoff
X
X    bsr.s    mandel
X
X    bsr  mouseon
X    bra  main
X
X*
X* Display portion of Mandelbrot set with top left corner a3+a4i
X* height and width a1*200
X*
X
Xmandel  movem.l  a1/a3/a4/a7,regstor
X    move #1,manding
X
X    move.l   $44e.w,a5
X
X    lea  200*160(a5),a6
X    move.l   a6,compare+2
X
X    move.l   a4,d4         bc=sy
X    moveq    #2-1,d0
Xrow2lp  move d0,-(sp)
X    move.l   a3,d3         ac=sx
X    bsr  calculate
X    lea  store1(pc),a2
X    move d1,(a2)+
X    add.l    a1,d3
X    bsr  calculate
X    move d1,(a2)+
X
X    move #318/2-1,d7
Xpairlp  add.l    a1,d3
X    move.l   d3,d2
X    add.l    a1,d3
X    bsr  calculate
X    move d1,2(a2)
X    cmp  -2(a2),d1
X    beq.s    same
```

```
X    exg  d3,d2
X    bsr  calculate
X    move d1,(a2)
X    move.l   d2,d3
X    bra.s    nxt1
Xsame    move d1,(a2)
Xnxt1    addq.l   #4,a2
X    dbra d7,pairlp
X
X    lea  store1(pc),a2
X    moveq    #20-1,d0
Xwordlp   move d0,-(sp)
X    moveq    #16-1,d2
Xbitlp    move (a2)+,d1
X    lsr  #1,d1
X    roxl #1,d0
X    lsr  #1,d1
X    roxl #1,d5
X    lsr  #1,d1
X    roxl #1,d6
X    lsr  #1,d1
X    roxl #1,d7
X    dbra d2,bitlp
X    move d0,(a5)+
X    move d5,(a5)+
X    move d6,(a5)+
X    move d7,(a5)+
X    move (sp)+,d0
X    dbra d0,wordlp
X
X    add.l    a1,d4
X    move (sp)+,d0
X    dbra d0,row2lp
X
X    lea  store2(pc),a2
X    lea  store1(pc),a4
X*
X* Calculate lines in pairs
X*
X
Xpair2lp  movem.l  d4/a2/a4,-(sp)
X    add.l    a1,d4
X
X    move.l   a3,d3         ac=sx
X    bsr  calculate
X    move d1,(a2)+
X    add.l    a1,d3
X    bsr  calculate
X    move d1,(a2)+
X
X    move #318/2-1,d7
Xpairlp0  add.l    a1,d3
X    move.l   d3,d2
```

```
X    add.l    a1,d3
X    bsr  calculate
X    move d1,2(a2)
X    cmp  -2(a2),d1
X    beq.s    same0
X    exg  d3,d2
X    bsr  calculate
X    move d1,(a2)
X    move.l   d2,d3
X    bra.s    nxt10
Xsame0   move d1,(a2)
Xnxt10   addq.l   #4,a2
X    dbra d7,pairlp0
X
X    move.l   4(sp),a2
X    move.l   a5,a6
X    lea  160(a5),a5
X    moveq    #20-1,d0
Xwordlp0  move d0,-(sp)
X    moveq    #16-1,d2
Xbitlp0   move (a2)+,d1
X    lsr  #1,d1
X    roxl #1,d0
X    lsr  #1,d1
X    roxl #1,d5
X    lsr  #1,d1
X    roxl #1,d6
X    lsr  #1,d1
X    roxl #1,d7
X    dbra d2,bitlp0
X    move d0,(a5)+
X    move d5,(a5)+
X    move d6,(a5)+
X    move d7,(a5)+
X    move (sp)+,d0
X    dbra d0,wordlp0
X
X    move.l   (sp)+,d5
X    movem.l (sp),a2/a4
X    move.l   d4,-(sp)
X    move.l   d5,d4
X    move.l   a3,d3
X    moveq    #20-1,d0
Xwordlp1  move d0,-(sp)
X    moveq    #16-1,d2
Xbitlp1   move (a2)+,d1
X    cmp  (a4)+,d1
X    beq.s    same1
X    bsr.s    calculate
Xsame1   add.l    a1,d3
X    lsr  #1,d1
X    roxl #1,d0
X    lsr  #1,d1
```

```
X    roxl #1,d5
X    lsr  #1,d1
X    roxl #1,d6
X    lsr  #1,d1
X    roxl #1,d7
X    dbra d2,bitlp1
X    move d0,(a6)+
X    move d5,(a6)+
X    move d6,(a6)+
X    move d7,(a6)+
X    move (sp)+,d0
X    dbra d0,wordlp1
X
X    movem.l  (sp)+,d4/a2/a4
X    exg  a2,a4
X    add.l    a1,d4
X
Xcompare  cmp.l    #0,a5
X    bne  pair2lp
X
Xmandout  clr  manding
X    movem.l  regstor(pc),a1/a3/a4/a7
X    rts           done mandelbrot
X
X
X* Calculate point at d3+d4i
X
Xcalculate
X    movem.l  d0/d2-d7,-(sp)
X
X    move.l   d3,d5
X    move.l   d4,d6
X    sub.l    a0,a0
X
Xcalc     move.l   d5,d2
X    bsr  square
X    bmi.s    out_of_range
X    move.l   d2,(temp).w
X    move.l   d6,d2
X    bsr  square
X    bmi.s    out_of_range
X    move.l   (temp).w,d1
X    add.l    d2,d1
X    bcs.s    done
X    cmp.l    #$80000000,d1
X    bcc.s    out_of_range
Xiterations
X    cmp.l    #100,a0
X    beq.s    done
X    add.l    d6,d6
X    bsr  multiply
X    add.l    d4,d6
X    move.l   (temp).w,d5
```

```
X    sub.l    d2,d5
X    add.l    d3,d5
X    addq #1,a0
X    bra.s    calc
X
Xout_of_range
X    moveq    #0,d1
X    move a0,d1
X    divu #15,d1
X    swap d1
X    addq #1,d1
X    movem.l  (sp)+,d0/d2-d7
X    rts
X
Xdone    moveq    #0,d1
X    movem.l  (sp)+,d0/d2-d7
X    rts
X
X* Square a number in d2
X
Xsquare  movem.l  d3/d4,-(sp)
X    tst.l    d2
X    bpl.s    splus
X    neg.l    d2
Xsplus   moveq    #0,d0
X    moveq    #0,d1
X    move.l   d2,d3
X    moveq    #0,d4
X
X    move d2,d3
X    move d2,d0
X    mulu d2,d0
X    swap d2
X    move d2,d1
X    mulu d2,d1
X    mulu d2,d3
X    swap d3
X    moveq    #0,d4
X    move d3,d4
X    clr  d3
X    add.l    d3,d0
X    addx.l   d4,d1
X    add.l    d3,d0
X    addx.l   d4,d1
X
X    rol.l    #3,d0
X    and  #7,d0
X    cmp.l    #$10000000,d1
X    bcc.s    overflow
X    lsl.l    #3,d1
X    or   d0,d1
X    move.l   d1,d2
X    movem.l  (sp)+,d3/d4
```

```
X    rts
X
Xoverflow
X    moveq    #-1,d0
X    movem.l  (sp)+,d3-d4
X    rts
X
X* Multiply d6 by d5
X
Xmultiply
X    movem.l  d2-d4,-(sp)
X
X    moveq    #0,d3
X    move.l   d5,d2
X
X    tst.l    d2
X    bpl.s    d2plus
X    moveq    #1,d3
X    neg.l    d2
Xd2plus  tst.l    d6
X    bpl.s    d6plus
X    eor  #1,d3
X    neg.l    d6
Xd6plus  moveq    #0,d0
X    moveq    #0,d1
X    moveq    #0,d4
X    move d3,-(sp)
X
X    move d2,d7
X    move d2,d0
X    mulu d6,d0
X    swap d2
X    move d2,d3
X    mulu d6,d3
X    swap d6
X    move d2,d1
X    mulu d6,d1
X    swap d3
X    moveq    #0,d4
X    move d3,d4
X    clr  d3
X    add.l    d3,d0
X    addx.l   d4,d1
X    mulu d6,d7
X    swap d7
X    move d7,d4
X    clr  d7
X    add.l    d7,d0
X    addx.l   d4,d1
X
X    rol.l    #3,d0
X    and  #7,d0
X    lsl.l    #3,d1
```

```
X    or   d0,d1
X    move.l    d1,d6
X    move (sp)+,d2
X    tst  d2
X    beq.s     mokay
X    neg.l     d6
Xmokay    movem.l   (sp)+,d2-d4
X    rts
X
X
X* Various routines
X
Xscreendown
X    move.l    phybase(pc),a0
X    lea  -1280(a0),a0
X    move #-1,-(sp)
X    pea  (a0)
X    pea  (a0)
X    move #5,-(sp)
X    trap #14
X    lea  12(sp),sp
X    rts
X
Xscreenup
X    move #-1,-(sp)
X    move.l    phybase(pc),-(sp)
X    move.l    (sp),-(sp)
X    move #5,-(sp)
X    trap #14
X    lea  12(sp),sp
X    rts
X
Xprhex32  move d0,-(sp)
X    swap d0
X    bsr.s     prhex16
X    move (sp)+,d0
Xprhex16  move d0,-(sp)
X    lsr  #8,d0
X    bsr.s     prhex8
X    move (sp)+,d0
Xprhex8   move d0,-(sp)
X    lsr  #4,d0
X    bsr.s     prhex4
X    move (sp)+,d0
Xprhex4   and  #$f,d0
X    move.b    hexdat(pc,d0),d0
X    bra.s     prchar
X
Xhexdat   dc.b '0123456789ABCDEF'
X
Xspace    moveq     #32,d0
Xprchar   move d0,-(sp)
X    pea  $30002
```

```
X    trap #13
X    addq.l   #6,sp
X    rts
X
Xprmes   pea  (a1)
X    move #9,-(sp)
X    trap #1
X    addq.l   #6,sp
X    rts
X
Xanykey  lea  anymes(pc),a1
X    bsr.s    prmes
X    pea  $20002
X    trap #13
X    addq.l   #4,sp
X    rts
X
Xsendikbd
Xkeylp   move.b   (a0)+,d1
X    bsr.s    sendkb
X    dbra d0,keylp
X    rts
X
Xsendkb  btst #1,$fc00.w
X    beq.s    sendkb
X    move.b   d1,$fc02.w
X    rts
X
Xwvbl    move.l   $466.w,d0
Xwaitlp  cmp.l    $466.w,d0
X    beq.s    waitlp
X    rts
X
Xmouseon  movem.l  d0-d7/a0-a6,-(sp)
X    bsr  drawbox
X    bsr  drawm
X    move.l   mousex(pc),nmousex
X    movem.l  (sp)+,d0-d7/a0-a6
X    move #1,mouse
X    rts
X
Xmouseoff
X    clr  mouse
X    movem.l  d0-d7/a0-a6,-(sp)
X    bsr  undraw
X    bsr  undrawbox
X    movem.l  (sp)+,d0-d7/a0-a6
X    rts
X
X
X* Vertical blank handler
X
Xvbl move keys(pc),d0
```

```
X
X    cmp  #$32,d0       m set multi-colours
X    bne.s    notm
X
X    movem.l  palette(pc),d0-d7
X    movem.l  d0-d7,$8240.w
X
Xnotm    cmp  #$1f,d0       s set up single colours
X    bne.s    nots
X
X    lea  $8242.w,a0
X    move.l   #$7770606,(a0)+
X    move.l   #$7770606,(a0)+
X    move.l   #$7770606,(a0)+
X    move.l   #$7770606,(a0)+
X    move.l   #$7770606,(a0)+
X    move.l   #$7770606,(a0)+
X    move.l   #$7770606,(a0)+
X    move #$777,(a0)
X
Xnots    cmp  #$2e,d0       c start cycling
X    bne.s    noc
X
X    move #1,cycle
X    move #5,tmpdel
X
Xnoc cmp  #$2f,d0       v stop cycling
X    bne.s    notsp
X
X    clr  cycle
X
Xnotsp   tst  cycle
X    beq.s    nocyc
X
X    cmp  #$4e,d0       + on keypad speed up cycling
X    bne.s    notdec
X
X    cmp  #1,delay
X    beq.s    notdec
X    subq #1,delay
X
Xnotdec   cmp  #$4a,d0       - on keypad slow down cycling
X    bne.s    notinc
X
X    addq #1,delay
X
Xnotinc   subq #1,tmpdel
X    bne.s    nocyc
X    move delay(pc),tmpdel
X    lea  $8260.w,a0
X    lea  $825e.w,a1
X    move (a1),d0
X    move -(a1),-(a0)
```

```
X    move -(a1),-(a0)
X    move -(a1),-(a0)
X    move -(a1),-(a0)
X    move -(a1),-(a0)
X    move -(a1),-(a0)
X    move -(a1),-(a0)
X    move -(a1),-(a0)
X    move -(a1),-(a0)
X    move -(a1),-(a0)
X    move -(a1),-(a0)
X    move -(a1),-(a0)
X    move -(a1),-(a0)
X    move -(a1),-(a0)
X    move d0,(a1)
X
Xnocyc   tst  mouse
X    beq  nomouse
X
X    bsr  undraw
X    bsr  undrawbox
X
X    move keys(pc),d0
X    move.l   nmousex(pc),d2
X    move.l   d2,d1
X    swap d1
X    cmp  #$48,d0
X    bne.s    notup
X    subq #1,d2
Xnotup    cmp  #$50,d0
X    bne.s    notdown
X    addq #1,d2
Xnotdown  cmp  #$4b,d0
X    bne.s    notleft
X    subq #1,d1
Xnotleft  cmp  #$4d,d0
X    bne.s    notrigh
X    addq #1,d1
X
Xnotrigh  tst  d1       make sure d1 & d2 are on screen
X    bpl.s    kmxok
X    clr  d1
Xkmxok    cmp  #319,d1
X    ble.s    kmxok1
X    move #319,d1
Xkmxok1   tst  d2
X    bpl.s    kmyok
X    clr  d2
Xkmyok    cmp  #199,d2
X    ble.s    kmyok1
X    move #199,d2
Xkmyok1   move d1,mousex
X    move d2,mousey
X
```

```
X    tst  leftbutton
X    beq.s    nolb
X    cmp  #1,box_width
X    beq.s    nolb
X    subq #1,box_width
Xnolb    tst  rightbutton
X    beq.s    norb
X    cmp  #100,box_width
X    beq.s    norb
X    addq #1,box_width
X
Xnorb    move box_width(pc),d5
X    moveq    #0,d6
X    move d5,d6
X    lsl  #3,d6
X    divu #5,d6
X    cmp  mousex(pc),d6
X    ble.s    mxokay
X    move d6,mousex
Xmxokay   cmp  mousey(pc),d5
X    ble.s    myokay
X    move d5,mousey
Xmyokay   bsr  drawbox
X    bsr  drawm
X    move.l   mousex(pc),nmousex
X
Xnomouse  rts
X
X
Xdrawm   moveq    #0,d0
X    move mousey(pc),d0
X    move d0,d1
X    add  d0,d0
X    add  d0,d0
X    add  d1,d0
X    lsl  #5,d0
X    move mousex(pc),d1
X    move d1,d2
X    lsr  #1,d1
X    and  #$fff8,d1
X    add  d1,d0
X    move.l   phybase(pc),a0
X    add.l    d0,a0
X    and  #$f,d2
X    lea  trash(pc),a1
X    move.l   mousedatapnt(pc),a2
X    move.l   a0,(a1)+
X    move #200,d3
X    sub  mousey(pc),d3
X    cmp  #16,d3
X    ble.s    fulsize
X    moveq    #16,d3
Xfulsize  subq #1,d3
```

```
X    move d3,(a1)+
X    cmp  #152,d1
X    beq.s    rightedge
Xdrawlp  movem.l  (a0),d4-d7
X    movem.l  d4-d7,(a1)
X    lea  16(a1),a1
X    moveq    #0,d0
X    move (a2)+,d0
X    not  d0
X    swap d0
X    lsr.l    d2,d0
X    not.l    d0
X    move.l   a0,a4
X    swap d0
X    and  d0,(a4)+
X    and  d0,(a4)+
X    and  d0,(a4)+
X    and  d0,(a4)+
X    swap d0
X    and  d0,(a4)+
X    and  d0,(a4)+
X    and  d0,(a4)+
X    and  d0,(a4)+
X    moveq    #0,d0
X    move (a2)+,d0
X    swap d0
X    lsr.l    d2,d0
X    or   d0,8(a0)
X    swap d0
X    or   d0,(a0)
X    moveq    #0,d0
X    move (a2)+,d0
X    swap d0
X    lsr.l    d2,d0
X    or   d0,10(a0)
X    swap d0
X    or   d0,2(a0)
X    lea  160(a0),a0
X    dbra d3,drawlp
X    rts
Xrightedge
X    movem.l  (a0),d4-d7
X    movem.l  d4-d7,(a1)
X    lea  16(a1),a1
X    move (a2)+,d0
X    not  d0
X    lsr  d2,d0
X    not  d0
X    move.l   a0,a4
X    and  d0,(a4)+
X    and  d0,(a4)+
X    and  d0,(a4)+
X    and  d0,(a4)+
```

```
X    move (a2)+,d0
X    lsr  d2,d0
X    or   d0,(a0)
X    move (a2)+,d0
X    lsr  d2,d0
X    or   d0,2(a0)
X    lea  160(a0),a0
X    dbra d3,rightedge
X    rts
X
Xundraw  lea  trash(pc),a0
X    move.l   (a0)+,a1
X    move (a0)+,d4
Xundrlp  movem.l  (a0)+,d0-d3
X    movem.l  d0-d3,(a1)
X    lea  160(a1),a1
X    dbra d4,undrlp
X    rts
X
Xundrawbox
X    move.l   oldbxy(pc),d1
X    bra.s    db
X
Xdrawbox  move.l   mousex(pc),d1
X    move.l   d1,oldbxy
Xdb  move.l   d1,d0
X    swap d0
X    move box_width(pc),d2
X    moveq    #0,d3
X    move d2,d3
X    lsl  #3,d3
X    divu #5,d3
X
X    move d1,d4
X    add  d1,d1
X    add  d1,d1
X    add  d4,d1
X    lsl  #5,d1
X    move.l   phybase(pc),a0
X    add  d1,a0
X    move d0,d7
X    and  #$f,d7
X    eor  #$f,d7
X    lsr  #1,d0
X    and  #$fff8,d0
X    add  d0,a0         a0=address of bottom right corner of box
X    move d2,d0
X    add  d0,d0
X    add  d0,d0
X    add  d2,d0
X    lsl  #5,d0
X    neg  d0
X    lea  0(a0,d0),a2
```

```
X    move d7,d6
X
X    moveq    #0,d1
Xhorizlp  bset d7,d1
X    subq #1,d3
X    bmi.s    donehor
X    addq #1,d7
X    cmp  #16,d7
X    bne.s    horizlp
X    moveq    #0,d7
X    eor  d1,(a0)
X    eor  d1,0(a0,d0)
X    moveq    #0,d1
X    subq.l   #8,a0
X    bra.s    horizlp
X
Xdonehor  eor  d1,(a0)
X    eor  d1,0(a0,d0)
X
X    add  d0,a0
X    cmp  #8,d7
X    bge.s    upby
X    addq.l   #1,a0
X    bra.s    chkd6
Xupby     and  #7,d7
Xchkd6    cmp  #8,d6
X    bge.s    upby1
X    addq.l   #1,a2
X    bra.s    vertlp
Xupby1    and  #7,d6
X
Xvertlp   lea  160(a0),a0
X    lea  160(a2),a2
X    subq #1,d2
X    beq.s    donever
X    bchg d7,(a0)
X    bchg d6,(a2)
X    bra.s    vertlp
X
Xdonever  rts
X
X
X* Keyboard handler
X
Xkeybrd   movem.l  d0/d1/a0,-(sp)
X
Xagain    move.b   $fc00.w,d1
X    btst #7,d1
X    beq  keyout
X    btst #0,d1
X    beq  keyo
X
X    moveq    #0,d0
```

```
X    move.b   $fc02.w,d0
X    tst  length
X    beq.s    nopack
X
X    move.l   bufpnt(pc),a0
X    move.b   d0,(a0)+
X    move.l   a0,bufpnt
X    subq #1,length
X    bne  keyo
X
X    lea  keybuf(pc),a0
X    and.b    #$fc,(a0)
X    cmp.b    #$f8,(a0)+
X    bne  keyo
X
X    move.b   (a0)+,d0
X    ext  d0
X    add  d0,nmousex
X    move.b   (a0)+,d0
X    ext  d0
X    add  d0,nmousey
X    bra  keyo
X
Xnopack   cmp  #$f6,d0
X    blt.s    akey
X
X    lea  keybuf(pc),a0
X    move.b   d0,(a0)+
X    move.l   a0,bufpnt
X    sub  #$f6,d0
X    move.b   lengths(pc,d0),d0
X    move d0,length
X    bra  keyo
X
Xlengths  dc.b 7,5,2,2,2,2,6,2,1,1
X
Xakey     cmp  #$52,d0       Test for mouse simulating keys
X    beq.s    leftb
X    cmp  #$74,d0
X    bne.s    notlon
Xleftb    move #1,leftbutton
X    bra  keyo
Xnotlon   cmp  #$d2,d0
X    beq.s    lefto
X    cmp  #$f4,d0
X    bne.s    notloff
Xlefto    clr  leftbutton
X    bra.s    keyo
Xnotloff  cmp  #$47,d0
X    beq.s    rightb
X    cmp  #$75,d0
X    bne.s    notron
Xrightb   move #1,rightbutton
```

```
X    bra.s     keyo
Xnotron   cmp  #$c7,d0
X    beq.s     righto
X    cmp  #$f5,d0
X    bne.s     notmkey
Xrighto   clr  rightbutton
X    bra.s     keyo
X
X* if not a mouse key then it's a keyboard key
X
Xnotmkey  tst.b     d0        Is it a on or off key
X    bmi.s     keyoff
X
X    move keys(pc),lastkey
X    move d0,keys
X    bra.s     keyo
X
Xkeyoff   and  #$7f,d0
X    cmp  keys(pc),d0   Is it current key going off
X    bne.s     notkeys
X
X    move lastkey(pc),keys   bring lastkey into current
X    clr  lastkey       current key off
X    bra.s     keyo
X
Xnotkeys  cmp  lastkey(pc),d0
X    bne.s     keyo
X
X    clr  lastkey       lastkey off
X
Xkeyo     move leftbutton(pc),d0
X    and  rightbutton(pc),d0
X    beq.s     noexm
X
X    tst  manding
X    beq.s     noexm
X
X    move.l    #mandout,14(sp)
X
Xnoexm    btst #5,d1
X    beq.s     keyout
X
X    tst.b     $fc02.w
X
Xkeyout   btst #4,$fa01.w
X    beq  again
X
X    movem.l   (sp)+,d0/d1/a0
X    bclr #6,$fa11.w
X    rte
X
X* Return to desktop
X
```

```
Xdesktop  move #$2700,sr
X    move.l   oldier(pc),$fa06.w
X    move.l   oldkeyb(pc),$118.w
X    move.l   $456.w,a0
X    move.l   oldvblq(pc),(a0)
X    move #$2300,sr
X
X    lea  keydat1(pc),a0
X    moveq    #2-1,d0
X    bsr  sendikbd
X
X    movem.l  oldpal(pc),d0-d7
X    movem.l  d0-d7,$8240.w
X
Xexit1   move oldres(pc),-(sp)
X    move.l   phybase(pc),-(sp)
X    move.l   logbase(pc),-(sp)
X    move #5,-(sp)
X    trap #14
X    lea  12(sp),sp
X    move.l   oldssp(pc),-(sp)
X    move #32,-(sp)
X    trap #1
X    addq.l   #6,sp
Xexit    clr  -(sp)
X    trap #1
X
X
X* Various data
X
Xnomem   dc.b 27,'EOut of memory',0
Xnomono  dc.b 27,'EColour monitors only',0
Xanymes  dc.b 13,10,13,10,'Press any key',0
Xinpmes  dc.b 27,'e',27,'H',27,'b',1,27,'K'
X    dc.b 'Enter number of iterations ',0
Xdelmes  dc.b 8,32,8,0
Xxmes    dc.b 27,'H',27,'K',27,'b',1,0
X
X    even
X
Xmousedatapnt
X    dc.l mousedata
Xmousedata
X    dc.w $7F,$FF80,0,$7F,$FF80,$7F00,$FF,$FF00
X    dc.w $7E00,$1FF,$FE00,$7C00,$3FF,$FC00,$7800,$1FF
X    dc.w $FE00,$7400,$8FF,$F700,$6200,$1C7F,$E380,$4100
X    dc.w $3E3F,$C1C0,$80,$FF1F,$E0,$40,$FF8F,$70
X    dc.w $20,$FFC7,$38,$10,$FFE3,$1C,8,$FFF1
X    dc.w 14,4,$FFFB,4,0,$FFFF,0,0
X
Xbox_width
X    dc.w 60
X
```

```
Xmouse    dc.w 0
Xkeys     dc.w 0
Xlastkey  dc.w 0
Xleftbutton   dc.w 0
Xrightbutton  dc.w 0
Xmousex   dc.w 120
Xmousey   dc.w 100
Xnmousex  dc.w 0
Xnmousey  dc.w 0
Xoldbxy   dc.l 0
Xlength   dc.w 0
Xbufpnt   dc.l 0
Xkeybuf   ds.b 8
Xkeydat   dc.b 7,4,8
Xkeydat1  dc.b 7,0
X
Xoldier   dc.l 0
Xoldkeyb  dc.l 0
Xoldssp   dc.l 0
Xoldvblq  dc.l 0
Xoldres   dc.w 0
Xphybase  dc.l 0
Xlogbase  dc.l 0
Xkeytable dc.l 0
X
Xbeepdat  dc.w $068,$104,$b00,$c49,$d00,$810,$7fe
X
Xpalette  dc.w $000,$072,$062,$153,$243,$234,$324,$415
X    dc.w $414,$413,$412,$500,$410,$320,$241,$062
X
Xcycle    dc.w 0
Xdelay    dc.w 5
Xtmpdel   dc.w 5
X
Xpattern  dc.w $ffff
X
Xmanding  dc.w 0
Xregstor  ds.l 4
Xlastmand
X    dc.l $a3d70a,$a0000000,$c0000000
X
Xinpbuf   ds.b 10
X
Xoldpal
Xbuffer   equ  oldpal+32
Xtrash    equ  buffer+512
Xstore1   equ  trash+300
Xstore2   equ  store1+640
Xend equ  store2+640
END_OF_FILE
if test 17097 -ne `wc -c <'MANDEL.S'`; then
    echo shar: \"'MANDEL.S'\" unpacked with wrong size!
fi
```

```
# end of 'MANDEL.S'
fi
echo shar: End of shell archive.
exit 0
```


```
From -4009258664494930359
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,2ff0b84730faa922,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-09-11 22:08:31 PST
Path: utzoo!attcan!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!
brutus.cs.uiuc.edu!apple!sun-barr!newstop!sun!grapevine!
koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i081: gfadiff -- File difference generator
Keywords: GFA-BASIC, shar
Message-ID: <34225@grapevine.uucp>
Date: 4 Sep 89 23:58:51 GMT
Sender: news@grapevine.EBay
Lines: 165
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: wscart01@ulkyvx.bitnet (Stuart Carter)
Posting-number: Volume 2, Issue 81
Archive-name: gfadiff
```


```
OK, i've seen some Diff programs posted here. Here is another.
This is the comparision program i made up to help me debug. All i ask
is that if changes are made, that the new code be uploaded to the
nearest BBS and that this code not be uploaded to the compu-serve
network. (Had a tiff with them 7 years ago, long story...)
Have fun with it....

Stuart

PS: One modification i ment to do but never did, the adress that the
    files Bload into should be even. It will speed up the process by
    50% load time.

#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      DIFF_315.LST
#
sed 's/^X//' << 'SHAR_EOF' > DIFF_315.LST &&
X'
X'    File Comparision Program by Stuart Carter  Version 3.15       c1988
X'
X'  I Have made this program avalible to the the public with the following
X'  rules:
```

```
X'
X'  1) Neither this work or any derivative of it will be sold for profit.
X'  2) This program may not be distibuted by or thru the Compu-Serve network.
X'  3) Any person Altering and uploading this code MUST:
X'       a)Add name and date to 'Modifying' author's list and
X'       b)Increment Version counter and
X'       c)Upload new modifed source code.
X'
X' Thats all...
X' Stuart Carter        (502)/491-0126       Bitnet: WSCART01@ULKYVX
X' Add Modifying authors names here...
X'
X'
X'
X'
X'
X'
X'
X1:
XT=5
XZ%=0
XC1%=1
XCls
XPrint At(23,1);"Comparision Program by Stuart Carter"
XPrint At(32,3);"NAME OF FILE #1"
XFileselect "*.*","",File1$
XIf File1$=""
X  End
XEndif
XPrint At(32,3);"Loading ......."
XOpen "R",#1,File1$,100
XField #1,100 As A$
XF1=Lof(#1)
XA=Int((F1)/100)+1
XDim F1$(A)
XFor X=1 To A-1
X  Get #1
X  F1$(X)=A$
XNext X
XClose #1
XIf F1<65000
X  Open "R",#1,File1$,1
X  Field #1,1 As A$
X  For Y=(X-1)*100 To F1
X    Get #1,Y
X    F1$(X)=F1$(X)+A$
X  Next Y
X  Close #1
XEndif
XPrint At(32,3);"NAME OF FILE #2"
XFileselect "*.*","",File2$
XIf File2$=""
X  End
```

```
XEndif
XPrint At(32,3);"Loading ......."
XOpen "R",#2,File2$,100
XField #2,100 As B$
XF2=Lof(#2)
XB=Int((F2)/100)+1
XDim F2$(B)
XFor X=1 To A-1
X  Get #2
X  F2$(X)=B$
XNext X
XClose #2
XIf F2<65000
X  Open "R",#2,File2$,1
X  Field #2,1 As B$
X  For Y=(X-1)*100 To F1
X    Get #2,Y
X    F2$(X)=F2$(X)+B$
X  Next Y
X  Close #2
XEndif
XPrint At(32,3);"                "
XPrint At(10,5);"file#1 : ";File1$;" is ";F1;" bytes"
XPrint At(10,7);"file#2 : ";File2$;" is ";F2;" bytes"
XPrint At(13,9);"Hexadecimal print(Y/N)";
XInput Z$
XIf Z$="Y" Or Z$="y" Or Z$="yes" Or Z$="YES" Then
X  H%=1
XEndif
XPrint At(13,11);"Send to printer(Y/N)";
XInput Z$
XIf Z$="Y" Or Z$="y" Or Z$="yes" Or Z$="YES" Then
X  Z%=1
XEndif
XCls
XIf Z%=1
X  Lprint "Comparision of ";File1$;" to ";File2$
XEndif
XPrint At(13,1);"Comparision of ";File1$;" to ";File2$
XPrint At(1,1);F1
XQ=3
XFor X=1 To A
X  If F1$(X)<>F2$(X)
X    Adf1=Varptr(F1$(X))
X    Adf2=Varptr(F2$(X))
X    For Y=0 To Len(F1$(X))
X      If Peek(Adf1+Y)<>Peek(Adf2+Y)
X        P1=(X-1)*100+Y
X        P2=Peek(Adf1+Y)
X        P3=Peek(Adf2+Y)
X        If H%=1
X          P1$=Hex$(P1)
X          P2$=Hex$(P2)
```

```
X          P3$=Hex$(P3)
X        Else
X          P1$=Str$(P1)
X          P2$=Str$(P2)
X          P3$=Str$(P3)
X        Endif
X        If Z%=1
X          Lprint P1$,P2$,P3$
X        Endif
X        Print At(10,Q);P1$,P2$,P3$
X        Q=Q+1
X      Endif
X      Print At(1,1);F1-(X-1)*100-Y;" "
X    Next Y
X  Endif
X  Print At(1,1);F1-(X-1)*100;"  "
XNext X
XQ=Q+1
XPrint At(1,1);"   "
XPrint At(1,Q)
XPrint At(1,Q);"PRESS RETURN TO CONTINUE"
XA=Inp(2)
XIf A<>13
X  End
XEndif
XClear
XGoto 1
SHAR_EOF
chmod 0600 DIFF_315.LST || echo "restore of DIFF_315.LST fails"
exit 0
```

```
From -6860825679507912987
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,c2065886da68ca68,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-09-24 22:10:29 PST
Path: utzoo!attcan!utgpu!jarvis.csri.toronto.edu!rutgers!cs.utexas.edu!uunet!lll-
winken!sun-barr!newstop!sun!grapevine!koreth%panarthea.ebay@sun.com
From: koreth%panarthea.ebay@sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i082: nolock -- Disable Caps-Lock key
Keywords: 68000
Message-ID: <34308@grapevine.uucp>
Date: 17 Sep 89 15:06:25 GMT
Sender: news@grapevine.EBay
Lines: 164
Approved: koreth%panarthea.ebay@sun.com

Submitted-by: dbrooks@osf.org (David Brooks)
Posting-number: Volume 2, Issue 82
Archive-name: nolock
```

```
*****************************************************************
*                                       *
*    NOLOCK                                  *
*                                       *
* After this program is executed, the CapsLock key will have
     *
* no effect, provided the keyboard translation table is set
     *
* only by calls to Keytbl() and Bioskeys().  You can still set
     *
* the tables by setting pointers into the vector whose address
     *
* is returned by Keytbl().                        *
*                                       *
* Written for jas (Sozobon assembler).  Thanks to Abacus's
     *
* "Atari ST Internals" for insights.                   *
*                                       *
* Copyright (c) 1989, David Brooks, All Rights Reserved.  Use
     *
* and distribute freely.                          *
*                                       *
*****************************************************************
*
*
* Some constant definitions (trap codes mainly)
*

GEMDOS        equ  1
BIOS     equ  13
XBIOS         equ  14
Keytbl        equ  16
Bioskeys  equ  24
Setexec       equ  5
Keep     equ  $31

XBIOSV        equ  46
XBIOSVAD  equ  $b8

sbit     equ  13       * Supervisor status bit

Magicnum  equ  $44424e4c* 'DBNL' must be just before trapad
*
* Program starts here.  It's all one text section, as we need to
* intermix code and data.  The initialization is at the end, and
* will be handed back to GEMDOS.
*

start:        bra  setup

*
* Static data:
*
```

```
transp:       ds.l 1        * Will point to table ptr vector
normp:        ds.l 1        * Default non-shift table
shiftp:       ds.l 1        * Default shifted table
trap14:       ds.l 1        * Replaced xbios vector

magic:        dc.l Magicnum

*
* Here's the xbios trap
*

trapad:
          lea  6(a7),a0 * Point to args if in super mode
          btst #sbit,(a7)
          bne  checkcodes
          dc.w $4e68         * move.l  USP,a0 -- Point to args
                        *  if in user mode
checkcodes:
          cmpi.w   #Bioskeys,(a0)
          beq  dobioskeys
          cmpi.w   #Keytbl,(a0)+
          beq  dokeytbl

          move.l   trap14,a0* Go to standard trap handler
          jmp  (a0)
*
* One of these routines for Bioskeys or Keytbl
*
dobioskeys:
          move.l   transp,a1* No args, no return.  Just
          move.l   normp,d0 *  set the table to norm, shift, norm
          move.l   d0,(a1)+
          move.l   shiftp,(a1)+
          move.l   d0,(a1)
          rte

dokeytbl:
          move.l   transp,d0* Three args, returns address of vector
          move.l   d0,a1
          move.l   (a0)+,d1
          bmi  nostd
          move.l   d1,(a1)       * put arg1 into standard and capslock
          move.l   d1,8(a1)
nostd:
          move.l   (a0),d1       * put arg2 into shift
          bmi  noshift
          move.l   d1,4(a1)
noshift:
          rte
keeptop:               * Top of kept memory

*
```

```
* Setup code.  The stack will be at tpa, and can be somewhat abused.
* 4(sp) points to the basepage.  All of this can be done in user
* mode.
*

setup:
         move.l   4(sp),a4
*
* First we look for our magic number.
*
         moveq.l  #-1,d5
         move.l   d5,(sp)
         move.w   #XBIOSV,-(sp)
         move.w   #Setexec,-(sp)
         trap #BIOS         * Setexec(46, -1L)
         move.l   d0,a0         * Check current vector
         cmpi.l   #Magicnum,-(a0)
         beq  quickexit* Already installed

*
* Locate the original (actually, the current) key table vector and the two
* interesting pointers.  d5 has -1L (see above).
*
         move.l   d5,(sp)
         move.l   d5,-(sp)
         move.l   d5,-(sp)
         move.w   #Keytbl,-(sp) * addr = Keytbl(-1L, -1L, -1L)
         trap #XBIOS
         move.l   d0,transp* Save the vector pointer
         move.l   d0,a0
         move.l   (a0)+,d1  * Save the normal table address
         move.l   d1,normp
         move.l   (a0)+,shiftp  * Save the shift table address
         move.l   d1,(a0)       * Set capslock = normal
*
* Insinuate our code in the trap 14 vector
*
         pea  trapad
         move.w   #XBIOSV,-(sp)
         move.w   #Setexec,-(sp)
         trap #BIOS         * Setexec(46, trapad)
         move.l   d0,trap14* Save old vector
*
* Exit, keeping the active code
*
         lea  keeptop,a0
         sub.l    a4,a0         * keeptop - basepage
         clr.w    (sp)
         move.l   a0,-(sp)
         move.w   #Keep,-(sp)
         trap #GEMDOS       * Keep(nbytes, 0)

*
```

```
* Come here if already installed for immediate exit
*
quickexit:
         clr.w    (sp)
         trap #GEMDOS

         end


From 5688358990511710712
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,4e3882dfdeb1b108,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-01 11:43:55 PST
Path: utzoo!utgpu!attcan!uunet!ginosko!brutus.cs.uiuc.edu!apple!sun-barr!newstop!
sun!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i083: builds09 -- Build Motorola S-records from .ttp files
Keywords: shar, C
Message-ID: <34418@grapevine.uucp>
Date: 1 Oct 89 18:43:55 GMT
Sender: news@grapevine.EBay
Lines: 357
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: johan@motstk.se (Johan Hagman)
Posting-number: Volume 2, Issue 83
Archive-name: builds09

These are the source files of builds 0.9, a program which creates
Motorola S-records from Atari ST binary (.ttp) files.

---
Johan Hagman           Motorola Computer Systems, Dalv. 2, 17136 Solna, Sweden
Phone: +46 8 7348800   uucp:  ..uunet!mcvax!sunic!motstk!johan
Fax:   +46 8 7354473   email: johan@motstk.{UUCP,SE}

#! /bin/sh
# This is a shell archive, meaning:
# 1. Remove everything above the #! /bin/sh line.
# 2. Save the resulting text in a file.
# 3. Execute the file with /bin/sh (not csh) to create:
#    README
#    builds.c
#    makefile
# This archive created: Thu Sep 28 21:49:10 1989
export PATH; PATH=/bin:/usr/bin:$PATH
echo shar: "extracting 'README'" '(951 characters)'
if test -f 'README'
then
     echo shar: "will not over-write existing file 'README'"
else
```

```
sed 's/^X//' << \SHAR_EOF > 'README'
X
X        builds version 0.9  17-Sep-89
X        ----------------------------
X
XThis program was developed to allow downloading code/data to a target
X68000 system via the serial port using the Motorola S-record format.
X
XBuilds produces S2-record files from ST .ttp executables. (There are
Xthree different S-record formats; S1, S2 and S3, for 16-, 24- and 32-bit
Xaddresses, respectively.)
X
XUsage:  builds [-address] [-v] file
X
X    "address" is the (24-bit) load- and execution address, and
X    should be entered in hex format WITHOUT a leading "0x". The
X    code is  relocated in the same way as the ST does when loading
X    a program. "address" defaults to zero.
X
X    "v" displays the program version.
X
XThe S-record file is written to stdout to allow redirection to a file
Xor the serial port using your favourite shell.
X
XCompiled with Sozobon C.
X
X    Johan Hagman  Motorola Computer Systems, Stockholm, Sweden
X             e-mail:  johan@motstk.{uucp,se}
X             uucp:    ..uunet!mcvax!sunic!motstk!johan
SHAR_EOF
if test 951 -ne "`wc -c < 'README'`"
then
     echo shar: "error transmitting 'README'" '(should have been 951 characters)'
fi
fi
echo shar: "extracting 'builds.c'" '(6157 characters)'
if test -f 'builds.c'
then
     echo shar: "will not over-write existing file 'builds.c'"
else
sed 's/^X//' << \SHAR_EOF > 'builds.c'
X/*
X * @(#)builds.c  0.9  17-Sep-89
X *
X *  Builds Motorola S-records from an Atari ST object file.
X *
X *  Copyright (c) 1989 Johan Hagman.
X *
X *  This program may be freely distributed, provided it retains
X *  my copyright notice. It is provided as is, with no warranty
X *  expressed or implied.
X *
X *  History:
```

```
X *
X *  Ver   Date    Author    Comments
X *  ---  -------  ------    -------------------
X *  0.9  22-Aug-88      J.H Initial coding.
X *       17-Sep-89          Last edit.
X */
X
X#include <stdio.h>
X#include <ctype.h>
X
X#define VERS  "0.9 17-Sep-89"
X#define OBJMAGIC 0x601A         /* magic number of ST object file */
X#define  SRECL     28      /* # of data bytes in each S-record */
X
Xtypedef unsigned char BOOLEAN;
X
Xtypedef struct {
X    WORD magic;
X    LONG tsiz;         /* text length */
X    LONG dsiz;         /* data length */
X    LONG bsiz;         /* bss length */
X    LONG unused[3];
X    WORD unusedToo;
X} HEADER;
X
Xstatic char  *myName;
Xstatic LONG  relocAddr = 0L;        /* relocation address */
Xstatic HEADER    hdr;
X
X
X/*---------------------------------------
X *
X *  readHeader -- open object file, read
X *  and check header.
X */
Xstatic FILE *readHeader(fname)
Xchar *fname;
X{
X    FILE *fd;
X
X    if ((fd = fopen(fname, "rb")) == NULL) {
X         fprintf(stderr, "%s: cannot open \"%s\"\n", myName, fname);
X         exit(1);
X    }
X    if (fread(&hdr, sizeof(HEADER), 1, fd) != 1)
X         perror("fread");
X    if (hdr.magic != OBJMAGIC) {
X         fprintf(stderr, "%s: \"%s\" is not an object file\n",
X              myName, fname);
X         exit(1);
X    }
X    return(fd);
X}
```

```
X
X
X/*---------------------------------------
X *
X *  getEntry -- returns relocation offset in *val,
X *  advancing *rltpos to next entry. Returns -1L in
X *  *val at end of table.
X */
Xstatic void getEntry(fd, rltpos, val)
XFILE *fd;
XLONG *rltpos;
XLONG *val;
X{
X    static    BOOLEAN first = TRUE;
X    register int  c;
X    LONG  streamOffset;
X
X    if (fgetpos(fd, &streamOffset) != 0)  /* store current offset */
X        perror("fgetpos");
X
X    if (fseek(fd, *rltpos, 0) != 0)       /* seek to relocation table */
X        c = 0;                   /* no table: PIC */
X    else
X        do {
X         if ((c = fgetc(fd)) == EOF) {
X             c = 0;              /* end of table */
X             break;
X         }
X         (*rltpos)++;            /* advance to next entry */
X
X         if (first) {            /* first entry is special */
X             first = FALSE;
X             *val += (LONG)c << 8;
X             c = 1;              /* loop once more */
X         } else
X             if (c == 1)
X              *val += 0xfeL;          /* offset exceeds 0xff */
X
X        } while (c == 1);
X
X    if (fsetpos(fd, &streamOffset) != 0)  /* return to where we were */
X        perror("fsetpos");
X
X    if (c == 0)
X        *val = -1L;                 /* end of table */
X    else
X        *val += (LONG)c;            /* accumulated rel. addr */
X}
X
X
X/*---------------------------------------
X *
X *  putSrec -- outputs a data record (S1 or S2).
```

```
X */
Xstatic void putSrec(type, addr, len, data)
Xchar type;
XLONG addr;
Xint len;
XBYTE data[];
X{
X    register BYTE cs;           /* checksum */
X    register int  i;
X
X    /* Calculate record length */
X    cs = len + ((addr == -1) ? 1 : (type == '1' ? 3 : 4));
X    printf("S%c%02X", type, cs);
X
X    if (addr != -1)                 /* suppress addr if -1 */
X         switch(type) {
X         case '1':              /* S1 record */
X              printf("%04lX", addr);
X              cs += (addr >> 8) + addr;
X              break;
X         case '2':              /* S2 record */
X              printf("%06lX", addr);
X              cs += (addr >> 16) + (addr >> 8) + addr;
X         /* S3 records not applicable for the 68k */
X         }
X
X    for (i = 0; i < len; i++) {
X         cs += data[i];
X         printf("%02X", data[i] & 0xff);  /* binary data */
X    }
X    printf("%02X\n", ~cs & 0xff);             /* checksum */
X}
X
X
X/*---------------------------------------
X *
X *  buildSrec -- creates the S-record file, adjusting absolute
X *  (32-bit) addresses according to user's relocation offset.
X */
Xstatic void buildSrec(fname)
Xchar *fname;
X{
X    BYTE dlist[SRECL];
X    int      input,        /* must keep track of two input */
X         nextInput;    /*  values at each iteration */
X    LONG rlval = 0L,
X         rltpos,       /* relocation table position in file */
X         bytesToDo,    /* # bytes in text + data section */
X         SrecAddr,
X         curOffset;    /* used for relocation */
X    register int col; /* "column" value for S-record data */
X    register LONG idx;/* index into text/data sections */
X    FILE *fd;
```

```
X
X    fd = readHeader(fname);              /* get object file header */
X
X    /* calculate relocation table offset */
X    rltpos = sizeof(HEADER) + hdr.tsiz + hdr.dsiz + 2;
X
X    getEntry(fd, &rltpos, &rlval);       /* read relocation entry */
X
X    if (fseek(fd, (long)sizeof(HEADER), 0) != 0)  /* position at text */
X    perror("fseek");
X
X    putSrec('0', -1L, strlen(fname), fname);  /* the header record */
X
X    bytesToDo = hdr.tsiz + hdr.dsiz;
X    SrecAddr = relocAddr;
X
X    if ((input = fgetc(fd)) == EOF)      /* read first byte in file */
X    perror("fgetc");
X
X    col = 0;
X    for (idx = curOffset = 0L; idx < bytesToDo; idx++) {
X
X    if ((nextInput = fgetc(fd)) == EOF)  /* get byte from file */
X        perror("fgetc");
X
X    if (idx == rlval) {
X
X        /* Oops! The next four bytes must be relocated */
X
X        getEntry(fd, &rltpos, &rlval);   /* find next relocation entry */
X
X        curOffset = relocAddr;       /* define relocation offset */
X    }
X
X    dlist[col] = (BYTE)(input + (curOffset >> 24));
X
X    if (curOffset != 0L)
X        /* we're currently relocating, now check for overflow */
X        if (nextInput + (BYTE)(curOffset >> 16) > 0xff)
X         dlist[col]++;           /* overflow */
X
X    input = nextInput;     /* advance one byte */
X    curOffset <<= 8;       /* shift in next less signif. byte */
X
X    if (++col == SRECL || idx == bytesToDo - 1) {
X
X        /* End of line or end of data - print the record */
X
X        putSrec('2', SrecAddr, col, dlist);
X        col = 0;                /* begin new record */
X        SrecAddr += SRECL;           /*   with a new address */
X    }
X    }
```

```
X    putSrec('9', -1L, 0);
X    fclose(fd);
X}
X
X
Xmain(argc, argv)
Xint argc;
Xchar *argv[];
X{
X    char *opt;                  /* arg. ptr */
X    LONG strtoul();
X
X    myName = "builds";
X
X    while (--argc > 0 && (*++argv)[0] == '-') {
X    opt = &argv[0][1];
X    switch (*opt) {
X    case 'v':
X        printf("%s v%s\n", myName, VERS);
X        exit(0);
X    default:
X        if (isxdigit(*opt)) {        /* hexadecimal? */
X         relocAddr = strtoul(opt, NULL, 16);
X         break;
X        }
X        fprintf(stderr, "%s: illegal option \"%c\"\n", myName, *opt);
Xusage:      fprintf(stderr, "usage: %s [-hexOffset] [-v] objFile\n", myName);
X        exit(0);
X    }
X    }
X
X    if (argc == 0)         /* must have a filename */
X    goto usage;
X
X    buildSrec(argv[0]);
X}
SHAR_EOF
if test 6157 -ne "`wc -c < 'builds.c'`"
then
     echo shar: "error transmitting 'builds.c'" '(should have been 6157
characters)'
fi
fi
echo shar: "extracting 'makefile'" '(126 characters)'
if test -f 'makefile'
then
     echo shar: "will not over-write existing file 'makefile'"
else
sed 's/^X//' << \SHAR_EOF > 'makefile'
X#
X#  A makefile for builds
X#
XCCOPT   = -O
```

```
XFILES   = builds.c
XBIN = builds.ttp
X
X$(BIN):      $(FILES)
X        cc $(CCOPT) -o $(BIN) $(FILES)
SHAR_EOF
if test 126 -ne "`wc -c < 'makefile'`"
then
     echo shar: "error transmitting 'makefile'" '(should have been 126
characters)'
fi
fi
exit 0
#    End of shell archive
```

```
From -8121794117822790112
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,ff2cf86c88ec2e70,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-01 11:55:59 PST
Path: utzoo!utgpu!attcan!uunet!ginosko!gem.mps.ohio-state.edu!apple!sun-barr!
newstop!sun!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i084: colorca -- Cellular automata in color
Keywords: shar, 68000
Message-ID: <34420@grapevine.uucp>
Date: 1 Oct 89 18:55:59 GMT
Sender: news@grapevine.EBay
Lines: 213
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: clong@topaz.rutgers.edu (Chris Long)
Posting-number: Volume 2, Issue 84
Archive-name: colorca

Here's a little CA (cellular automaton) program written by me and
a friend many moons ago.  I've included one version for color low-res,
and another for color high-res (better known as low- and med-res,
respectively).  I've also included my desktop.inf, which seems to set
the colors rather nicely for the med-res version.  Have fun!  If you
liked this program, let me know.

BTW, let it run.  Just when you think it's reached a steady-state,
it'll do something completely unexpected.

#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      COLORCAH.S
#      COLORCAL.S
```

```
#
sed 's/^X//' << 'SHAR_EOF' > COLORCAH.S &&
X* A color one-dimensional cellular automata program by Chris Long and
X* John Balsley.  Program is old (this was the second program we wrote
X* for the ST; the first was ye typical "hello" program), and can probably
X* be speeded up and improved with a miniscule amount of thought and
X* effort.  Designed to be used in high-res; particularly nice against
X* a black background.
X
Xintin   equ     8
Xptsin   equ     12
X
Xinit    equ     $a000
Xsetpix  equ     $a001
Xgetpix  equ     $a002
X
X    dc.w init
X    move.l   intin(a0),a3
X    move.l   ptsin(a0),a4
X
X    lea.l    frame1,a5
X    move.b   #1,(a5)
X    move.b   #1,201(a5)
X    clr  cycles
X
Xnxtscr  lea.l    frame1,a5
X    lea.l    frame2,a6
X    adda.l   #1,a6
X
X    move cycles,(a4)
X    move #0,2(a4)
X    clr  d5
X
Xnxtcll1 move.b   (a5)+,d5
X    add.b    1(a5),d5
X
X    cmpi #15,d5
X    blt  alive1
X
X    move.b   #0,(a6)+
X    move #0,(a3)
X    dc.w $a001
X    addq #1,2(a4)
X    cmpi #200,2(a4)
X    bne  nxtcll1
X    jmp  out1
X
Xalive1  move.b   d5,(a6)+
X    move d5,(a3)
X    dc.w $a001
X    addq #1,2(a4)
X    cmpi #200,2(a4)
X    bne  nxtcll1
```

```
X
Xout1    addq #1,cycles
X    lea.l    frame2,a5
X    lea.l    frame1,a6
X    adda.l   #1,a6
X
X    move cycles,(a4)
X    move #0,2(a4)
X    clr  d5
X
Xnxtcll2  move.b   (a5)+,d5
X    add.b    1(a5),d5
X
X    cmpi #15,d5
X    blt  alive2
X
X    move.b   #0,(a6)+
X    move #0,(a3)
X    dc.w $a001
X    addq #1,2(a4)
X    cmpi #200,2(a4)
X    bne  nxtcll2
X    jmp  out2
X
Xalive2   move.b   d5,(a6)+
X    move d5,(a3)
X    dc.w $a001
X    addq #1,2(a4)
X    cmpi #200,2(a4)
X    bne  nxtcll2
X
Xout2    addq #1,cycles
X    cmpi #640,cycles
X    bne  nxtscr
X    clr  cycles
X    jmp  nxtscr
X
Xcycles   ds.w 1
Xframe1   ds.b 202
Xframe2   ds.b 202
X
X    end
SHAR_EOF
chmod 0600 COLORCAH.S || echo "restore of COLORCAH.S fails"
sed 's/^X//' << 'SHAR_EOF' > COLORCAL.S &&
X* A color one-dimensional cellular automata program by Chris Long and
X* John Balsley.  Program is old (this was the second program we wrote
X* for the ST; the first was ye typical "hello" program), and can probably
X* be speeded up and improved with a miniscule amount of thought and
X* effort.  Designed to be used in low-res.
X
Xintin   equ     8
Xptsin   equ     12
```

```
X
Xinit    equ     $a000
Xsetpix  equ     $a001
Xgetpix  equ     $a002
X
X    dc.w init
X    move.l   intin(a0),a3
X    move.l   ptsin(a0),a4
X
X    lea.l    frame1,a5
X    move.b   #1,(a5)
X    move.b   #1,201(a5)
X    clr  cycles
X
Xnxtscr  lea.l    frame1,a5
X    lea.l    frame2,a6
X    adda.l   #1,a6
X
X    move cycles,(a4)
X    move #0,2(a4)
X    clr  d5
X
Xnxtcll1 move.b   (a5)+,d5
X    add.b    1(a5),d5
X
X    cmpi #15,d5
X    blt  alive1
X
X    move.b   #0,(a6)+
X    move #0,(a3)
X    dc.w $a001
X    addq #1,2(a4)
X    cmpi #200,2(a4)
X    bne  nxtcll1
X    jmp  out1
X
Xalive1   move.b   d5,(a6)+
X    move d5,(a3)
X    dc.w $a001
X    addq #1,2(a4)
X    cmpi #200,2(a4)
X    bne  nxtcll1
X
Xout1    addq #1,cycles
X    lea.l    frame2,a5
X    lea.l    frame1,a6
X    adda.l   #1,a6
X
X    move cycles,(a4)
X    move #0,2(a4)
X    clr  d5
X
Xnxtcll2  move.b   (a5)+,d5
```

```
X    add.b    1(a5),d5
X
X    cmpi #15,d5
X    blt  alive2
X
X    move.b   #0,(a6)+
X    move #0,(a3)
X    dc.w $a001
X    addq #1,2(a4)
X    cmpi #200,2(a4)
X    bne  nxtcll2
X    jmp  out2
X
Xalive2  move.b   d5,(a6)+
X    move d5,(a3)
X    dc.w $a001
X    addq #1,2(a4)
X    cmpi #200,2(a4)
X    bne  nxtcll2
X
Xout2    addq #1,cycles
X    cmpi #320,cycles
X    bne  nxtscr
X    clr  cycles
X    jmp  nxtscr
X
Xcycles  ds.w 1
Xframe1  ds.b 202
Xframe2  ds.b 202
X
X    end
SHAR_EOF
chmod 0600 COLORCAL.S || echo "restore of COLORCAL.S fails"
exit 0


From -2424549108419328307
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,f8966ee6535c7bdf,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-01 12:01:37 PST
Path: utzoo!utgpu!attcan!uunet!ginosko!gem.mps.ohio-state.edu!apple!sun-barr!
newstop!sun!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i085: eternal3 -- Reset-proof ramdisk
Keywords: 68000
Message-ID: <34421@grapevine.uucp>
Date: 1 Oct 89 19:01:37 GMT
Sender: news@grapevine.EBay
Lines: 446
Approved: koreth@panarthea.ebay.sun.com
```

Submitted-by: ntomczak@ualtavm.bitnet (Michal Jaegermann)
Posting-number: Volume 2, Issue 85
Archive-name: eternal3

   Recently Volker A. Brandt (VBRANDT@DBNUAMA1.BITNET) posted in
comp.sys.atari.st binaries for a nice, small, sweet and fast reset proof
ram-drive which works also on Mega 4. But these were only binaries.
For all of you who would like to 'roll your own' here is a source
of similar ram-drive which I am using, with a great success, for quite
a while. Please check details of operation in a header comment.

   This code works also with Rainbow TOS without any problems.

   If you are going to enter 'ram-drives demolition derby' then you
may want to recode a read-write loop rw_l1 using 'moveml.' trick. You
will be probably able to measure a difference, though I do not know
if you will notice it. Happy hacking...

   Michal Jaegermann
   Myrias Research Corporation
   Edmonton, Alberta, CANADA
   mj@myrias.COM

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;here I come;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
********
*
*   etrnl3/4.s - a version of resizable reset-proff ram-drive
*                modified to work also on Mega4
*
*   An original of code eternal by John Franco.
*   This code modifies and fixes some bugs of eternal2.
*
*   A 'fractional' name of this program - eternal 3 for Mega 4 -
*   follows a fine tradition originated by one operating system.
*
*   Changes to work on Mega4 - by Michal Jaegermann, 23 May 1988
*    this code also somehow modified in other places
*
*   Assemble this source with as68.
*
*   An unmodified source will produce a ram-drive which will be
*   installed as a drive m: with size of 300k, unless you have
*   Mega 4, in which case a default size will be 1M.
*
*   This program, when executed first time, will set a number of pointers
*   and it will reset your computer.  Execute the second time in order
*   to install ram-drive.  To automate that process, put, as the
*   FIRST program in AUTO folder.  If, on a reset,  an old ram-drive is found,
*   and no keys will intervene, then its contents will be, hopefully,
*   restored.
*
*   If some other program plays the same games with phystop pointer
*   as this program, then etrnl3/4 will be not installed, unless you
```

```
*   will force it by pressing an 'Alternate' key.  In that case, too bad
*   for the other program.
*
*   If a 'Control' key is held down, then ram-drive will be not installed.
*   If a ram drive was installed - it will be unmounted and a computer will
*   reset.  A message wil appear and it will stay until you hit a
*   'Return' key. If this program is in AUTO folder, hit 'Control' once
*   again, in order to prevent an installation of default ram-drive
*   during boot-up sequence.
*
*   'Alternate' key will let you specify drive letter and size of a ram
*   drive to install.  Already installed ram-drive will be replaced
*   with a new one.  During this process an old contents of the ram-drive
*   is wiped out.  Even if a drive letter and size remain unchanged!
*   An example of a ram drive specification - 'p200'.  Your input
*   is not checked, so be careful. Do not use drive letters already taken,
*   unless resizing ram-drive previously mounted by etrnl3/4.prg
*
*   NOTE:  For ram-drives installed with sizes around 2 MBytes and over
*   some version of TOS will come out with a wrong size information when
*   inquired from Desktop.  'df' function in gulaam fares much better.
*   If somebody has an explanation for this behaviour I am all ears.
*   It is really not very well tested when amounts of memory allocated
*   to ram-drive are in eccess of 3 MBytes.
*
*******

*  Section for an assembler to produce some constants
*  No code generated

     .offset 0

offrecsiz:    ds.w 1
offcls:       ds.w 1
offclsb:  ds.w 1
offrdlen: ds.w 1
offfsiz:  ds.w 1
offfatrec:    ds.w 1
offdatrec:    ds.w 1
offnumcl: ds.w 1
offbflg:  ds.w 1

offident: ds.b 4
offdrvno: ds.w 1

magic         .equ 'rdrv'
reset_v       .equ 4
phystop       .equ $42e
membot        .equ $432
memtop        .equ $436
hdv_bpb       .equ $472
hdv_rw        .equ $476
hdv_mediach   .equ $47e
```

```
drivebits .equ $4c4
conterm       .equ $484

gemdos        .equ $01
mshrink       .equ $4a
super         .equ $20
necin         .equ $07
printline .equ $09
readline  .equ $0a
bios      .equ $0d
kbshift       .equ $0b

ram_drive .equ 12   ; drive m: as68 does not constants like 'm'-'a'
st_size       .equ 300  ; default ramdisk size in K's
mega_size .equ 1024 ; default ramdisk size for Mega 4

     .text

     movea.l   a7,a5              ; standard preamble
     movea.l   #stack,a7
     movea.l   $4(a5),a5
     move.l    $C(a5),d0
     add.l     $14(a5),d0
     add.l     $1C(a5),d0
     add.l     #$100,d0
     move.l    d0,-(a7)
     move.l    a5,-(a7)
     move.w    #0,-(a7)
     move.w    #mshrink,-(a7)
     trap #gemdos
     adda.l    #$C,a7

     clr.l     -(a7)
     move.w    #super,-(a7)
     trap #gemdos
     addq.l    #6,a7
     move.l    d0,sv_sup     ; save old supervisor pointer
     bclr #$0,conterm
     move.l    memtop,a4
     move.l    #phystop,a5
        moveq   #0,d3
     cmpi.l  #$400000,(a5)
        beq     mega4
     movea.l   (a5),a5
     lea  offident(a5),a1
     move.l    (a1),d3       ;I if drive installed - magic in d3
     bra  keychk
mega4:
     movea.l   (a5),a5            ; a 5now contains a value of phystop
     move.w    #mega_size,(default+4)  ; modify default size for mega4
keychk:
     move.w    #$FFFF,-(a7)
     move.w    #kbshift,-(a7)
```

```
     trap #bios
     addq.l   #4,a7
     tst  d0
     beq  nokey

     tst  d3       ; no ram-drive if 0
        beq   chkalt

*    you are here if something above (false) phystop and some key pushed


     cmpi.l   #magic,d3 ; is this ram drive
     bne.b    chkalt        ; if no - then check which key it was

*    de-install if not 'Alternate'

     lea  offdrvno(a5),a3
     move.w   (a3)+,d4
     move.w   drivebits,d5
     bclr d4,d5
     move.w   d5,drivebits
     add.b    d4,rmlet  ; set up drive letter in a message
     move.l   (a3)+,phystop
     move.l   (a3)+,memtop
     addq.l  #2,a3
     move.l   (a3)+,hdv_bpb
     addq.l  #2,a3
     move.l  (a3)+,hdv_rw
     addq.l  #2,a3
     move.l  (a3),hdv_mediach

chkalt:
     lsr.l    #4,d0
     bcs  getspec       ; if Alt was pressed - ask specs
     cmpi.l   #magic,d3 ; do we have an old ram drive?
     bne  exit      ; no? - get out and do nothing

     move.l   #rmmess,-(a7)
     move.w   #printline,-(a7)
     trap #gemdos
     addq.l   #6,a7
waitcr:  move.w  #necin,-(a7)
     trap #gemdos
     addq.l   #2,a7
     cmpi.b   #$0d,d0       ; cr?
     bne.b    waitcr        ; try again if not
     bra  reset

getspec:
     move.l   #askspec,-(a7)
J#Imove.w#printline,-(a7)
     trap #gemdos
     addq.l   #6,a7
```

```
     move.l   #inpbuf,-(a7) ; read answer here
     move.w   #$A,-(a7)
     trap #$1
J#Iaddq.l#6,a7
     moveq    #0,d1
     move.b  entered,d1
     tst.b    d1
     beq  getspec

     moveq    #0,d0
     move.b   drvlet,d0
     subq.l   #1,d0
     andi.l   #$1F,d0
     move.w   d0,device_no
     move.b   d0,rd_bpb+3
     move.b   d0,rd_rw+3
     move.b   d0,rd_mediach+3

     cmpi.b   #$1,d1
     ble.b    default
     cmpi.b   #$6,d1
     ble.b    convert
     move.b   #$5,d1
convert:
     move d1,d6
     subq.l   #2,d6
     moveq    #0,d0
     movea.l  #capacity,a0
collect:
     mulu #$A,d0
     move.b   (a0)+,d1
     andi.l   #$F,d1
     add.l    d1,d0
     dbf  d6,collect
     addq.l   #2,d0
     bra.b    newsize

nokey:
     cmpi.l   #magic,d3
     beq.w    restart

default:
     move.l   #st_size,d0   ; this instruction modified if mega4 detected
newsize:
     move.w   d0,numcl
     moveq    #$A,d2
     lsl.l    d2,d0         ; compute size in bytes
     add.l    #$2600,d0 ; + drive overhead
     move.l   d0,d1
     add.l    membot,d1 ; + system
     add.l    #$20000,d1    ; minimum for user
     cmp.l    a4,d1         ; exit if not enough memory
     bge  exit
```

```
     move.l  #sv_phystop,a3  ; we may need old data in case of removal
     move.l   a5,(a3)+ ; save old phystop
     move.l   a4,(a3)+ ; save old memtop
     sub.l    d0,a5         ; decrease by a requested size
     sub.l    d0,a4
     move.l  a5,phystop ; and tell computer that this is it
     move.l  a4,memtop
     addq.l  #2,a3
     move.l  hdv_bpb,(a3)+   ; fill jump table
     addq.l  #2,a3      ; skip jump instruction in the table
     move.l  hdv_rw,(a3)+
     addq.l  #2,a3
     move.l   hdv_mediach,(a3)
     movea.l  #driver,a0
     moveq    #$7F,d7
movehi:
     move.l   (a0)+,(a5)+   ; move disk-driver into its place
     dbf  d7,movehi
     move.l   #$1100,d7
     moveq    #$0,d0
erase:
     move.l   d0,(a5)+ ; wipe-out FATs
     dbf  d7,erase
reset:
     movea.l  reset_v,a0
     jmp  (a0)      ; and force warm boot

restart:
     move.l   a5,d1              ; pointer to phystop
     add.l    #(rd_mediach-driver),d1 ; install new values in vectors
     move.l   d1,hdv_mediach
     add.l    #(rd_bpb-rd_mediach),d1
     move.l   d1,hdv_bpb
     add.l    #(rd_rw-rd_bpb),d1
     move.l   d1,hdv_rw

     move.w   offdrvno(a5),d1        ; put device on map
     move.w   drivebits,d0
     bset d1,d0
     move.w   d0,drivebits
     clr.w   offbflg(a5)
exit:
     move.l   sv_sup,-(a7)
     move.w   #super,-(a7)
     trap #gemdos
     addq.l   #6,a7
     clr.w    -(a7)              ; by for now
     trap #gemdos
     illegal                ; don't tread on me

* ------------------------------------------------------------
* this data moved into high memory
```

```
*
driver:
recsiz:  dc.w 512M
clsiz:   dc.w 2
clsizb: dc.w  1024
rdlen:   dc.w 7        ; root dir len in sectors
fsiz:    dc.w 5        ; FAT size in sectors
fatrec: dc.w  6        ; here starts 2nd FAT
datrec: dc.w  18       ; here data start
numcl:  dc.w  $01EC         ; number of clusters
bflags: dc.w  0

ident:
     dc.b 'rdrv'
device_no:
     dc.w ram_drive
sv_phystop:
     dc.l 0
sv_memtop:
     dc.l 0
jbpb:                  ; jump table to be filled
     dc.w $4ef9
vbpb:
     dc.l 0        ; space for vector to old bpb handler
jrw:
     dc.w $4ef9
vrw:
     dc.l 0        ; space for vector to old bpb handler
jmediach:
     dc.w $4ef9
vmediach:
     dc.l 0        ; space for vector to old bpb handler

rd_mediach:
     cmpi.w   #ram_drive,$4(a7)
     bne  jmediach
     moveq    #$0,d0
     rts

rd_bpb:
     cmpi.wI#ram_drive,$4(a7)
     bne  jbpb
     move.l   phystop,d0
     rts
rd_rw:
     cmpi.w   #ram_drive,$E(a7)
     bne  jrw
     movea.l  phystop,a0
     adda.l   #$200,a0
     movea.l  $6(a7),a1
     moveq    #$0,d1
     move.w   $C(a7),d1
     moveq    #$9,d0
```

```
     asl.l    d0,d1
     adda.l   d1,a0
     move.w   $A(a7),d0
     move.l   a1,d2
     btst #$0,d2
     bne.b    rw_l3
     btst #$0,$5(a7)
     bne.b    rw_l1
     exg  a0,a1
rw_l1:
     move.w   #$F,d1
rw_l2:
     move.l   (a1)+,(a0)+
     move.l   (a1)+,(a0)+
     move.l   (a1)+,(a0)+
     move.l   (a1)+,(a0)+
     move.l   (a1)+,(a0)+
     move.l   (a1)+,(a0)+
     move.l   (a1)+,(a0)+
     move.l   (a1)+,(a0)+
     dbf  d1,rw_l2
     subq.l   #1,d0
     bne.b    rw_l1
     rts

rw_l3:
     btst #$0,$5(a7)
     bne.b    rw_l4
     exg  a0,a1
rw_l4:
     move.w   #$3F,d1
rw_l5:
     move.b   (a1)+,(a0)+
     move.b   (a1)+,(a0)+
     move.b   (a1)+,(a0)+
     move.b   (a1)+,(a0)+
     move.b   (a1)+,(a0)+
     move.b   (a1)+,(a0)+
     move.b   (a1)+,(a0)+
     move.b   (a1)+,(a0)+
     dbf  d1,rw_l5
     subq.l   #1,d0
     bne.b    rw_l4
     rts

     .data
askspec:
     dc.b 'Enter ramdisk spec: ',0
inpbuf:
     dc.b 6
entered:
     dc.b 5
drvlet:
```

```
     dc.b 0
capacity:
     dc.b 0,0,0,0,0,0,0,0

rmmess:  dc.b 'Ram drive '
rmlet:   dc.b 'a'
rmcont: dc.b  ' unmounted. Reset...',0

     .even
     .bss
     ds.b $200
stack:  ds.l    1
sv_sup:
     ds.l 1
savstack:
     ds.l 1
     end
*
* Michal Jaegermann
*m j@myrias.COM
* (...uunet,alberta)!myrias!mj   -- uucp
*


From 7579071784269062016
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,4f75c1ef39472db,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-03 14:27:42 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!uwm.edu!gem.mps.ohio-state.edu!
apple!sun-barr!newstop!sun!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i086: degadesk -- Load DEGAS .P?3 files onto desktop
Keywords: shar, C, 68000, high
Message-ID: <34443@grapevine.uucp>
Date: 3 Oct 89 21:27:42 GMT
Sender: news@grapevine.EBay
Reply-To: kruysbergen@hnykun53.bitnet (Noud van Kruysbergen)
Lines: 1004
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: kruysbergen@hnykun53.bitnet (Noud van Kruysbergen)
Posting-number: Volume 2, Issue 86
Archive-name: degadesk

These are the sources to the DESKTOP1.UUE programs [posted to binaries -sg].

Greetings,

Noud van Kruysbergen
kruysbergen@hnykun53.bitnet
```

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      DESKTOP1.C
#      DESKTOP3.C
#      DESKTOP2.S
#
sed 's/^X//' << 'SHAR_EOF' > DESKTOP1.C &&
X; ************************************************************************
X; This is the source of DESKTOP.ACC, a program that changes the normal gray
X; background with a Degas P?3 file. This source is public domain, but I
X; would appreciate it very much if a program that uses (part) of this
X; source respects the name of the author!
X; ************************************************************************
X; This source is written for Megamax C, with the inline assembly.
X; ************************************************************************
X; Noud van Kruysbergen
X; N.I.C.I.
X; P.O. Box 9104
X; 6500 HE Nijmegen
X; The Netherlands
X; email: kruysbergen@hnykun53.bitnet
X; ************************************************************************
X#include <gemdefs.h>
X#include <stdio.h>
X#include <osbind.h>
X#include <gembind.h>
X
Xint status=1;
Xint handle;
Xint menu_id,msgbf[8];
Xint i,j;
Xlong l,picstart,bufcount,piccount;
X
Xstatic long _oldgem();
Xstatic long _newgem();
X
Xstatic int _style(),_interior(),_mode();
Xextern int gl_apid;
Xstatic long _buffer();
Xlong buffer;
X
Xstatic int _clipx0(),_clipy0(),_clipx1(),_clipy1();
Xchar path[50],file[14];
Xchar t[70];
X
Xchar newdta[44];
X
Xnewroutines()
X{
X     asm
X     {
```

```
X                   dc.l      0x58425241
X                   dc.l      0x4e5f4450
X                   dc.l      0L
X
X          _newgem: cmpi.w    #115,D0
X                   bne       gemquit
X                   movem.l   D0-A3,-(A7)
X
X                   move.l    D1,A0
X                   move.l    (A0),A0
X                   move.w    (A0),D0
X
X                   cmpi.w    #114,D0
X                   bne       nofill
X                   lea       _interior,A0
X                   move.w    (A0),D0
X                   cmpi.w    #2,D0
X                   bne       gemrest
X                   lea       _style,A0
X                   move.w    (A0),D0
X                   cmpi.w    #4,D0
X                   bne       gemrest
X
X                   lea       _mode,A0
X                   move.w    (A0),D0
X                   beq       gemrest
X                   move.l    D1,A0
X                   move.l    8(A0),A0
X                   move.w    (A0)+,D0
X                   bne       gemrest
X                   move.w    (A0)+,D0
X                   cmpi.w    #19,D0
X                   beq       menu
X                   tst.w     D0
X                   bne       gemrest
X          menu:    move.w    (A0)+,D0
X                   cmpi.w    #639,D0
X                   bne       gemrest
X                   move.w    (A0),D0
X                   cmpi.w    #399,D0
X                   bne       gemrest
X
X                   move.l    0x44e,A1
X; A1: screenadress
X                   move.w    _clipy0,D0
X                   move.w    D0,D2
X; D2: y0
X                   muls      #80,D0
X                   adda.l    D0,A1
X; A1: screenadress + y0 offset
X                   lea       _buffer,A0
X                   move.l    (A0),A0
X; A0: pictureadress
```

```
X                       adda.l    D0,A0
X; A0: pictureadress + y0 offset
X                       move.w    _clipy1,D0
X                       sub.w     D2,D0
X                       move.w    D0,D2
X; D2: total number of lines
X                       move.w    _clipx0,D3
X; D3: x0
X                       move.w    D3,D7
X                       ext.l     D7
X                       divs      #8,D7
X                       move.w    D7,D3
X                       adda.l    D3,A0
X                       adda.l    D3,A1
X; D3: x0/8
X                       swap      D7
X                       move.w    D7,D4
X; D4: x0%8
X                       move.w    _clipx1,D5
X                       addq.w    #1,D5
X; D5: x1
X                       move.w    D5,D7
X                       ext.l     D7
X                       divs      #8,D7
X                       move.w    D7,D5
X; D5: x1/8
X                       swap      D7
X                       move.w    D7,D6
X; D6: x1%8
X                       move.l    D5,D0
X                       sub.l     D3,D0
X                       bne       label3
X
X                       move.w    D6,D7
X                       subi.w    #8,D7
X                       neg.w     D7
X                       move.b    #-1,D3
X                       lsr.b     D7,D3
X                       lsl.b     D7,D3
X                       lsl.b     D4,D3
X                       lsr.b     D4,D3
X            loop2:     move.b    D3,D5
X                       move.b    (A0),D7
X                       and.b     D5,D7
X                       not.b     D5
X                       and.b     (A1),D5
X                       add.b     D5,D7
X                       move.b    D7,(A1)
X                       adda.l    #80,A0
X                       adda.l    #80,A1
X                       dbf       D2,loop2
X                       bra       selfok
X
```

```
X          label3:   subq.w    #1,D0
X                    tst.w     D4
X                    seq       D5
X                    beq       yloop
X                    tst.w     D0
X                    sne       D5
X                    beq       yloop
X                    subq.w    #1,D0
X; first incomplete byte
X          yloop:    move.l    A0,A2
X                    move.l    A1,A3
X                    tst.w     D4
X                    beq       nostart
X                    move.b    (A2)+,D3
X                    move.b    #-1,D7
X                    lsr.b     D4,D7
X                    and.b     D7,D3
X                    not.b     D7
X                    and.b     (A3),D7
X; D1 contains lowest picturebits, D7 highest backgroundbits
X                    add.b     D3,D7
X                    move.b    D7,(A3)+
X
X          nostart:  move.w    D0,D7
X                    tst.w     D5
X                    beq       endbyte
X          xloop:    move.b    (A2)+,(A3)+
X                    dbf       D7,xloop
X; last incomplete byte
X          endbyte:  tst.w     D6
X                    beq       no_end
X                    move.b    (A2)+,D3
X                    move.b    #-1,D7
X                    lsr.b     D6,D7
X                    not.b     D7
X                    and.b     D7,D3
X                    not.b     D7
X                    and.b     (A3),D7
X                    add.b     D3,D7
X                    move.b    D7,(A3)
X
X          no_end:   adda.l    #80,A0
X                    adda.l    #80,A1
X          label1:   dbf       D2,yloop
X
X          selfok:   movem.l   (A7)+,D0-A3
X                    rte
X
X          nofill:   move.l    D1,A0
X                    move.l    4(A0),A0
X                    move.w    (A0),D2
X                    cmpi.w    #23,D0
X                    bne       nointerior
```

```
X                    lea       _interior,A0
X                    move.w    D2,(A0)
X                    bra       gemrest
X
X          nointerior:cmpi.w   #24,D0
X                    bne       nostyle
X                    lea       _style,A0
X                    move.w    D2,(A0)
X                    bra       gemrest
X
X          nostyle:  cmpi.w    #129,D0
X                    bne       gemrest
X                    lea       _mode,A0
X                    move.w    D2,(A0)
X                    beq       gemrest
X                    move.l    D1,A0
X                    move.l    8(A0),A0
X                    lea       _clipx0,A1
X                    move.w    (A0)+,(A1)
X                    lea       _clipy0,A1
X                    move.w    (A0)+,(A1)
X                    lea       _clipx1,A1
X                    move.w    (A0)+,(A1)
X                    lea       _clipy1,A1
X                    move.w    (A0),(A1)
X
X          gemrest:  movem.l   (A7)+,D0-A3
X          gemquit:  move.l    _oldgem,-(A7)
X                    rts
X
X          _oldgem:  dc.l      0L
X          _buffer:  dc.l      0L
X          _style:   dc.w      0
X          _interior:dc.w      0
X          _mode:    dc.w      0
X          _clipx0:  dc.w      0
X          _clipy0:  dc.w      0
X          _clipx1:  dc.w      0
X          _clipy1:  dc.w      0
X     }
X}
X
XReadfile()
X{
X     handle=-1;
X     strcpy(t,path);
X     j=0;
X     while(t[j++]!='*' && j<70);
X     if (j<70)
X     {
X          path[j]=0;
X          t[--j]=0;
X          strcat(path,".P?3");
```

```
X          strcat(t,file);
X          if ((handle=Fopen(t,0))>=0)
X          {
X               Fseek(34L,handle,0);
X               j=strlen(file);
X               if (file[--j]=='3' && file[--j]=='C')
X               {
X                    l=Fsetdta(newdta);
X                    Fsfirst(t,0);
X                    Fsetdta(l);
X                    l=*(long *)(newdta+26);
X                    if (piccount=picstart=Malloc(l))
X                    {
X                         l=Fread(handle,l,picstart);
X                         bufcount=buffer;
X                         do
X                         {
X                              asm
X                              {
X                                             move.l    piccount(A4),A0
X                                             move.b    (A0),D0
X                                             ext.w     D0
X                                             and.w     #0xff,D0
X                                             move.w    D0,j(A4)
X                                             addq.l    #1,piccount(A4)
X                              }
X                              if (j<128)
X                              {
X                                   asm
X                                   {
X                                                  move.l    bufcount(A4),A2
X                                                  move.l    piccount(A4),A1
X                                                  move.w    j(A4),D0
X                                        pic1loop: move.b    (A1)+,(A2)+
X                                                  dbf       D0,pic1loop
X                                                  move.l    A1,piccount(A4)
X                                                  move.l    A2,bufcount(A4)
X                                   }
X                              }
X                              else
X                              {
X                                   if (j!=128)
X                                   {
X                                        j^=255;
X                                        asm
X                                        {
X                                                       move.l    bufcount(A4),A2
X                                                       move.l    piccount(A4),A1
X                                                       move.w    j(A4),D0
X                                                       addq.w    #1,D0
X                                             pic2loop: move.b    (A1),(A2)+
X                                                       dbf       D0,pic2loop
X                                                       addq.l    #1,piccount(A4)
```

```
X                                                       move.l    A2,bufcount(A4)
X                                             }
X                                        }
X                                   }
X                              }
X                              while((piccount-picstart<l) && (bufcount-
buffer<32000L));
X                              Mfree(picstart);
X                         }
X                         else form_error(8);
X                    }
X                    else Fread(handle,32000L,buffer);
X                    Fclose(handle);
X               }
X     }
X     return(handle);
X}
X
XInstall()
X{
X          asm
X          {
X                              lea       _oldgem,A0
X                              move.l    0x88,(A0)
X                              lea       _newgem,A1
X                              move.l    A1,0x88
X                              move.l    (A0),-4(A1)
X          }
X}
X
Xmain()
X{
X     appl_init();
X     buffer=Malloc(32000L);
X     asm
X     {
X                         lea       _buffer,A0
X                         move.l    buffer(A4),(A0)
X     }
X     strcpy(t,"A:\\DESKTOP.P?3");
X     strcpy(path,"A:\\*.P?3");
X     t[file[0]=0]=path[0]=(char)(Dgetdrv()+65);
X     Fsetdta(newdta);
X     i=Fsfirst(t,0);
X     if (!i)
X     {
X          strcpy(file,(char *)(newdta+30));
X          if (Readfile()<0) status=file[0]=0;
X          else Supexec(Install);
X     }
X     menu_id=menu_register(gl_apid,"  Desktop Picture");
X     while ()
X     {
```

```
X          evnt_mesag(msgbf);
X/*************************************************************
X--- for version 2.01: include these lines ---
X          if (status)
X          {
X               l=Super(0L);
X               asm
X               {
X                              lea       _newgem,A1
X                              cmpa.l    0x88,A1
X                              beq       already
X                              lea       _oldgem,A0
X                              move.l    0x88,(A0)
X                              move.l    A1,0x88
X                              move.l    (A0),-4(A1)
X                    already:
X               }
X               Super(l);
X          }
X*************************************************************/
X          if (msgbf[0]==AC_OPEN && msgbf[4]==menu_id)
X          {
X               i=form_alert(3-status,"[2][| Desktop Picture 2.01|= Noud van
Kruysbergen|][New| On |Off]");
X               if (i!=3-status)
X               {
X                    wind_update(BEG_UPDATE);
X                    form_dial(FMD_START,0,0,0,0,0,0,640,400);
X                    if ((i==2 && file[0]==0) || i==1)
X                    {
X                         fsel_input(path,file,&j);
X                         if (j && Readfile()<0)
X                         {
X                              form_error(2);
X                              file[0]=0;
X                         }
X                    }
X                    if (((i==2 && file[0]!=0) || i==3) && status!=(j=3-i))
X                    {
X                         if (status=j) Supexec(Install);
X                         else
X                         {
X                              l=Super(0L);
X                              asm
X                              {
X                                             lea       _oldgem,A0
X                                             move.l    (A0),0x88
X                              }
X                              Super(l);
X                         }
X                    }
X                    form_dial(FMD_FINISH,0,0,0,0,0,0,640,400);
X                    wind_update(END_UPDATE);
```

```
X               }
X          }
X     }
X}
X
SHAR_EOF
chmod 0600 DESKTOP1.C || echo "restore of DESKTOP1.C fails"
sed 's/^X//' << 'SHAR_EOF' > DESKTOP3.C &&
X; ************************************************************************
X; This is the source of DESKTOP.ACC, a program that changes the normal gray
X; background with a Degas P?3 file. This source is public domain, but I
X; would appreciate it very much if a program that uses (part) of this
X; source respects the name of the author!
X; ************************************************************************
X; This source is written for Megamax C, with the inline assembly.
X; ************************************************************************
X; This program needs DESKTOP.PRG version 1.0.
X; ************************************************************************
X; Noud van Kruysbergen
X; N.I.C.I.
X; P.O. Box 9104
X; 6500 HE Nijmegen
X; The Netherlands
X; email: kruysbergen@hnykun53.bitnet
X; ************************************************************************
X#include <gemdefs.h>
X#include <stdio.h>
X#include <osbind.h>
X#include <gembind.h>
X
Xint handle;
Xint menu_id,msgbf[8];
Xint i;
X
Xint clip[4]={0,0,639,399};
X
Xint contrl[12],intin[128],ptsin[128],intout[128],ptsout[128];
Xint work_out[57],work_in[11]={1,1,1,1,1,1,1,1,1,1,2};
X
Xextern int gl_apid;
Xextern long addr_out[2];
X
Xlong picture=0L;
Xlong l,picstart,bufcount,piccount;
X
Xchar newdta[44];
Xchar path[50],file[14];
Xchar t[70];
X
Xmain()
X{
X     appl_init();
X     v_opnvwk(work_in,&handle,work_out);
```

```
X     vs_clip(handle,-1,clip);
X     if (*(long *)intout==0x4e5f4450) picture=*(long *)(intout+4);
X     strcpy(path,"A:\\*.P?3");
X     file[0]=0;
X     path[0]=(char)(Dgetdrv()+65);
X     menu_id=menu_register(gl_apid,"  Desktop Picture");
X     while ()
X     {
X          evnt_mesag(msgbf);
X          if (msgbf[0]==AC_OPEN && msgbf[4]==menu_id)
X          {
X               if (picture==0L) form_alert(1,"[3][DESKTOP 1.1 not resident]
[Cancel]");
X               else
X               {
X                    i=form_alert(3,"[2][| Desktop Picture 2.03|(with DESKTOP.PRG
1.1)|= Noud van Kruysbergen|][P?3|Gray|Cancel]");
X                    if (i<3)
X                    {
X                         wind_update(BEG_UPDATE);
X                         form_dial(FMD_START,0,0,0,0,0,0,640,400);
X                         if (i==1)
X                         {
X                              fsel_input(path,file,&i);
X                              if (i)
X                              {
X                                   strcpy(t,path);
X                                   i=0;
X                                   while(t[i++]!='*' && i<70);
X                                   if (i<70)
X                                   {
X                                        path[i]=0;
X                                        t[--i]=0;
X                                        strcat(path,".P?3");
X                                        strcat(t,file);
X                                        if ((handle=Fopen(t,0))>=0)
X                                        {
X                                             Fseek(34L,handle,0);
X                                             i=strlen(file);
X                                             if (file[i-2]=='C')
X                                             {
X                                                  l=Fsetdta(newdta);
X                                                  Fsfirst(t,0);
X                                                  Fsetdta(l);
X                                                  l=*(long *)(newdta+26);
X                                                  if
(piccount=picstart=Malloc(l))
X                                                  {
X
l=Fread(handle,l,picstart);
X                                                       bufcount=picture;
X                                                       do
X                                                       {
```

```
X                                                                   asm
X                                                                   {
X                                                                        move.l
piccount(A4),A0
X                                                                        move.b
(A0),D0
X                                                                        ext.w
D0
X                                                                        and.w
#0xff,D0
X                                                                        move.w
D0,i(A4)
X                                                                        addq.l
#1,piccount(A4)
X                                                                   }
X                                                                   if (i<128)
X                                                                   {
X                                                                        asm
X                                                                        {
X
move.l    bufcount(A4),A2
X
move.l    piccount(A4),A1
X
move.w    i(A4),D0
X                                                                             pic1loop:
move.b    (A1)+,(A2)+
X
dbf       D0,pic1loop
X
move.l    A1,piccount(A4)
X
move.l    A2,bufcount(A4)
X                                                                        }
X                                                                   }
X                                                                   else
X                                                                   {
X                                                                        if (i!=128)
X                                                                        {
X                                                                             i^=255;
X                                                                             asm
X                                                                             {
X
move.l    bufcount(A4),A2
X
move.l    piccount(A4),A1
X
move.w    i(A4),D0
X
addq.w    #1,D0
X
pic2loop: move.b    (A1),(A2)+
X
```

```
dbf       D0,pic2loop
X
addq.l    #1,piccount(A4)
X
move.l    A2,bufcount(A4)
X                                                                    }
X                                                               }
X                                                          }
X                                                     }
X                                                     while((piccount-
picstart<l) && (bufcount-picture<32000L));
X                                                     Mfree(picstart);
X                                                }
X                                                else form_error(8);
X                                           }
X                                           else Fread(handle,32000L,picture);
X                                           Fclose(handle);
X                                      }
X                                      else
X                                      {
X                                           form_error(2);
X                                           file[0]=0;
X                                      }
X                                 }
X                            }
X                            else
X                            {
X                                 asm
X                                 {
X                                           move.l    picture(A4),A0
X                                           move.l    A0,A1
X                                           move.w    #199,D1
X                                 loop1:    adda.l    #80,A1
X                                           move.w    #79,D0
X                                 loop2:    move.b    #0x55,(A0)+
X                                           move.b    #0xaa,(A1)+
X                                           dbf       D0,loop2
X                                           adda.l    #80,A0
X                                           dbf       D1,loop1
X                                 }
X                            }
X                            form_dial(FMD_FINISH,0,0,0,0,0,0,640,400);
X                            wind_update(END_UPDATE);
X                       }
X                  }
X             }
X     }
X}
X
SHAR_EOF
chmod 0600 DESKTOP3.C || echo "restore of DESKTOP3.C fails"
sed 's/^X//' << 'SHAR_EOF' > DESKTOP2.S &&
```

```
X; ***************************************************************************
X; This is the source of DESKTOP.PRG, a program that changes the normal gray
X; background with a Degas PI3 file. This source is public domain, but I
X; would appreciate it very much if a program that uses (part) of this
X; source respects the name of the author!
X; ***************************************************************************
X; This specific version (1.1) returns the picture adress in intout[2]
X; when vs_clip() is called. "N_DP" is returned in intout[0] to
X; be able to check whether this adress is valid.
X; ***************************************************************************
X; Noud van Kruysbergen
X; N.I.C.I.
X; P.O. Box 9104
X; 6500 HE Nijmegen
X; The Netherlands
X; email: kruysbergen@hnykun53.bitnet
X; ***************************************************************************
X
X         text
X
X         move.l    4(A7),A0          ; these lines are the normal
X         move.l    #$100,D4          ; ones for a TSR program
X         add.l     12(A0),D4         ; there's nothing interesting
X         add.l     20(A0),D4         ; here.
X         add.l     28(A0),D4         ; You only have to add 32000 bytes
X         add.l     #32000,D4         ; for the (PI3) picture.
X
X         lea       noudtext,A0       ; show the copyright lines
X         bsr       cconws            ; to the screen
X
X         clr.w     -(A7)             ; try to open the file
X         pea       file1             ; "DESKTOP.PI3" in the AUTO
X         move.w    #$3d,-(A7)        ; folder.
X         trap      #1                ; D3 = Fopen("AUTO\DESKTOP.PI3",0);
X         addq.l    #8,A7
X         move.w    D0,D3
X         bpl       ok1               ; File exists: goto ok1
X
X         clr.w     -(A7)             ; If the file does not exist:
X         pea       file2             ; try to open "DESKTOP.PI3" in the
X         move.w    #$3d,-(A7)        ; root directory.
X         trap      #1                ; D3 = Fopen("DESKTOP.PI3",0);
X         addq.l    #8,A7
X         move.w    D0,D3
X         bpl       ok1               ; File exist: goto ok1
X
X         lea       error1,A0         ; No "DESKTOP.PI3" exists
X         bsr       cconws            ; print this to screen
X         move.w    #7,-(A7)          ; let the bell ring
X         move.w    #2,-(A7)
X         trap      #1                ; Cconout(7);
X         addq.l    #4,A7
X         bsr       wait              ; wait for a moment to be able to
```

```
X          clr.w     -(A7)              ; read the message and
X          trap      #1                 ; Pterm0();
X
Xok1:      clr.w     -(A7)              ; The file exists. Skip the
X          move.w    D3,-(A7)           ; PI3 header info (34 bytes).
X          move.l    #34,-(A7)
X          move.w    #$42,-(A7)
X          trap      #1                 ; Fseek(34L,D3,0);
X          adda.l    #10,A7
X
X          pea       picbuffer          ; Read the picture in the
X          move.l    #32000,-(A7)       ; buffer (which begins at
X          move.w    D3,-(A7)           ; 'picbuffer'.
X          move.w    #$3f,-(A7)
X          trap      #1                 ; Fread(D3,32000L,picbuffer);
X          adda.l    #12,A7
X          cmpi.l    #32000,D0          ; 32000 bytes read?
X          beq       ok2                ; goto ok2
X          lea       error2,A0          ; else print warning to screen
X          bsr       cconws
X          bsr       wait               ; wait, but do install!
X
Xok2:       pea       gemnew             ; the new gem vector
X          move.w    #34,-(A7)
X          move.w    #5,-(A7)
X          trap      #13                ; gemold=Setexc(34,gemnew);
X          addq.l    #8,A7
X          move.l    D0,gemold
X          lea       gemnew,A0
X          move.l    D0,-4(A0)          ; return adress for XBRA protocol
X
X          move.w    D3,-(A7)           ; close the file
X          move.w    #$3e,-(A7)
X          trap      #1                 ; Fclose(D3);
X          addq.l    #4,A7
X
X          clr.w     -(A7)              ; terminate and stay resident
X          move.l    D4,-(A7)           ; with no errorcode
X          move.w    #$31,-(A7)
X          trap      #1                 ; Pterm(D4,0);
X
X; ---------------------------------------------------------------------------
X
Xcconws:    move.l    A0,-(A7)           ; routine to print line to screen
X          move.w    #9,-(A7)
X          trap      #1                 ; Cconws(A0);
X          addq.l    #6,A7
X          rts
X
Xwait:      move.w    #1000,D1           ; routine that eats up CPU time
Xwait1:     move.w    #1000,D0           ; simply to wait
Xwait2:     dbf       D0,wait2
X          dbf       D1,wait1
```

```
X          rts
X
X; ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
X; this is where it all is about. The new gemvector starts here.
X; ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
X
X          dc.b      "XBRAN_DP"
X          dc.l      0
Xgemnew:    cmpi.w    #115,D0             ; test whether it is a VDI call
X          bne       gemquit
X
X; ************************************************************************
X; It is a VDI call!
X
X          movem.l   D0-A3,-(A7)         ; save registers
X
X          move.l    D1,A0
X          move.l    (A0),A0             ; D0 = control[0];
X          move.w    (A0),D0             ; what VDI call is it?
X
X; ************************************************************************
X; Is the VDI call vr_recfl()?
X
X          cmpi.w    #114,D0
X          bne       nofill
X          cmpi.w    #2,interior         ; Has vsf_interior(handle,2);
X          bne       gemrest             ; been called?
X          cmpi.w    #4,style            ; Has vsf_style(handle,4);
X          bne       gemrest             ; been called?
X
X          tst.w     mode                ; Is it in the clipping mode?
X          beq       gemrest
X          move.l    D1,A0
X          move.l    8(A0),A0            ; D0 = intin[0] etc..
X          move.w    (A0)+,D0            ; x0==0
X          bne       gemrest
X          move.w    (A0)+,D0            ; y0==0 || y0==19
X          cmpi.w    #19,D0
X          beq       menu
X          tst.w     D0
X          bne       gemrest
Xmenu:      move.w    (A0)+,D0            ; x1==639
X          cmpi.w    #639,D0
X          bne       gemrest
X          move.w    (A0),D0             ; y1==399
X          cmpi.w    #399,D0
X          bne       gemrest
X
X; All the conditions are true. They mean that a gray background should
X; be drawn. The exact size is set in the clipping variables. They are
X; used to draw the background.
X
X          move.l    $44e,A1             ; A1 = *(long *)0x44e;
```

```
X          move.w    clipy0,D0
X          move.w    D0,D2               ; D0 = D2 = clipy0;
X          muls      #80,D0              ; D0 *= 80 /* 80 characters per line */
X          adda.l    D0,A1               ; A1 = screenadres + clipy0 offset;
X          lea       picbuffer,A0        ; A0 = pictureadres;
X          adda.l    D0,A0               ; A0 = pictureadres + clipy0 offset;
X          move.w    clipy1,D0           ; D0 = clipy1;
X          sub.w     D2,D0               ; D0 = clipy1 - clipy0;
X          move.w    D0,D2               ; D2 = total number of lines;
X          move      clipx0,D3           ; D3 = x0;
X          move.w    D3,D7
X          ext.l     D7
X          divs      #8,D7
X          move.w    D7,D3
X          adda.l    D3,A0
X          adda.l    D3,A1               ; D3 = clipx0/8;
X          swap      D7
X          move.w    D7,D4               ; D4 = clipx0%8;
X          move.w    clipx1,D5
X          addq.w    #1,D5
X          move.w    D5,D7
X          ext.l     D7
X          divs      #8,D7
X          move.w    D7,D5               ; D5 = clipx1/8;
X          swap      D7
X          move.w    D7,D6               ; D6 = clipx1%8;
X          move.l    D5,D0
X          sub.l     D3,D0
X          bne       label3              ; if (D3>D5) goto label3;
X
X; ---------------------------------------------------------------------------
X; This routine takes care of the times when the desktop has to be redrawn
X; within one character (for instance from bit 2 to bit 5).
X; ---------------------------------------------------------------------------
X
X          move.w    D6,D7               ; D7 = D6; /* D6=5+1 */
X          subi.w    #8,D7
X          neg.w     D7                  ; D7 = -(D6-8); /* D7=2 */
X          move.b    #-1,D3              ; D3 = 11111111 /* in binary mode */
X          lsr.b     D7,D3               ; D3 = 00111111
X          lsl.b     D7,D3               ; D3 = 11111100
X          lsl.b     D4,D3               ; D3 = 11110000 /* D4=2 */
X          lsr.b     D4,D3               ; D3 = 00111100
X
X; This is a very elegant way to make a byte that has the bits set that
X; should contain the background information. A zero in this byte means that
X; what is on screen at that location should stay there. In this case only
X; the bits 5,4,3 and 2 (numbers 2,3,4 and 5 from the left) have to be updated.
X
Xbyteloop: move.b    D3,D0               ; D0 = 00111100
X          move.b    (A0),D7             ; D7 = picturebyte
X          and.b     D0,D7               ; D7 &= 00111100
X          not.b     D0                  ; D0 = 11000011
```

```
X          and.b     (A1),D0             ; screenbyte &= 11000011
X          add.b     D0,D7               ; D7 = picturebyte + screenbyte
X          move.b    D7,(A1)             ; screenbyte = D7
X          adda.l    #80,A0              ; next pictureline
X          adda.l    #80,A1              ; next screenline
X          dbf       D2,byteloop         ; do this for all # lines
X          bra       selfok              ; quit the GEM call
X
X; ---------------------------------------------------------------------------
X; More than one byte have to be redrawn.
X; ---------------------------------------------------------------------------
X
Xlabel3:    subq.w    #1,D0               ; # bytes -= 1 for 'dbf' sake.
X          tst.w     D4                  ; xclip0 on a byte-boundairy?
X          seq       D5                  ; yes : set D5
X          beq       yloop               ;     : goto yloop
X          tst.w     D0                  ; D0==0? /* only 1 byte? */
X          sne       D5                  ; no  : set D5
X          beq       yloop               ; yes : goto yloop
X          subq.w    #1,D0               ; more than one byte: D0 -= 1;
X
X; ---------------------------------------------------------------------------
X; Start of the actual updating routine.
X; ---------------------------------------------------------------------------
X
X; We first have to draw the remainder of the first byte. If there is no
X; remainder (D4=0) we don't have to do this. Suppose we have to redraw
X; from bit 4 (from the left, so actually bit 5): D4=3.
X
Xyloop:     move.l    A0,A2               ; A2 = pictureadres
X          move.l    A1,A3               ; A3 = screenadres
X          tst.w     D4                  ; if no remainder has to be drawn
X          beq       nostart             ; then don't do this
X          move.b    (A2)+,D3            ; D3 = picturebyte
X          move.b    #-1,D7              ; D7 = 11111111
X          lsr.b     D4,D7               ; D7 = 00011111
X          and.b     D7,D3               ; D3 &= 00011111 /* picturebits */
X          not.b     D7                  ; D7 = 11100000
X          and.b     (A3),D7             ; D7 &= screenbyte /* screenbits */
X          add.b     D3,D7               ; D7 += D3 /* picture+screen */
X
X; D7 contains the high bits from the old background and the low bits from
X; the redrawn background.
X
X          move.b    D7,(A3)+            ; screenbyte = D7
X
X; When we arrived here, we are at a byte boundary. We are going to copy
X; bytes from the picture to the screen if necessary (D5 was set)
X
Xnostart:   move.w    D0,D7               ; D7 = # bytes
X          tst.w     D5
X          beq       endbyte             ; D5 not set: goto endbyte
Xxloop:     move.b    (A2)+,(A3)+         ; screenbyte = picturebyte
```

```
X          dbf       D7,xloop            ; do this # bytes times
X
X; We still are at a byte boundary. Are there any bits to draw any further?
X
Xendbyte:  tst.w     D6
X          beq       no_end
X
X; If not, skip this part. D6 contains the number of bits to update, say 3.
X
X          move.b    (A2)+,D3            ; D3 = picturebyte
X          move.b    #-1,D7              ; D7 = 11111111
X          lsr.b     D6,D7               ; D7 = 00011111
X          not.b     D7                  ; D7 = 11100000
X          and.b     D7,D3               ; D3 &= 11100000 /* picturebits */
X          not.b     D7                  ; D7 = 00011111
X          and.b     (A3),D7             ; D7 &= screenbyte
X          add.b     D3,D7               ; D7 = screenbits + picturebits
X          move.b    D7,(A3)             ; screenbyte = D7
X
Xno_end:   adda.l    #80,A0              ; next picture line
X          adda.l    #80,A1              ; next screen line
Xlabel1:   dbf       D2,yloop            ; do this # lines times
X
Xselfok:   movem.l   (A7)+,D0-A3         ; quit the GEM call
X          rte
X
X; ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
X; This is were the redrawing of the background ends. It all is in fact
X; nothing more than shifting a proper amount of bits at a proper time!
X; ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
X
X; ************************************************************************
X; Is the VDI call vst_interior()?
X
Xnofill:   move.l    D1,A0
X          move.l    4(A0),A0
X          move.w    (A0),D2             ; D2 = intin[0];
X
X          cmpi.w    #23,D0
X          bne       nointerior
X          move.w    D2,interior         ; interior = intin[0];
X          bra       gemrest
X
X; ************************************************************************
X; Is the VDI call vst_style()?
X
Xnointerior:cmpi.w   #24,D0
X          bne       nostyle
X          move.w    D2,style            ; style = intin[0];
X          bra       gemrest
X
X; ************************************************************************
X; Is the VDI call vs_clip()?
```

```
X
Xnostyle:  cmpi.w    #129,D0
X          bne       gemrest
X          move.w    D2,mode             ; mode = intin[0];
X          beq       gemrest             ; if (mode==0) no clipping;
X
X          cmpi.w    #-1,D2
X          bne       normal
X
X; ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
X; for version 1.0 exclude the part between the ...++++++... lines !!!
X;
X; We return the picture adress in intout[2] when vs_clip(x,-1,x) is called.
X; This is an illegal not documented feature! When DESKTOP.PRG is not
X; installed the content of intout[2] is NOT the picture adress!
X; This can be controlled by intout[0]: if *(long *)intout==$4e5f4450
X; ("N_DP") the value *(long *)(intout+4) is a valid picture adress.
X
X          move.l    D1,A0
X          move.l    12(A0),A0
X          move.l    #$4e5f4450,(A0)+    ; *(ong *)intout=0x4e5f4450
X          clr.l     (A0)+
X          lea       picbuffer,A1
X          move.l    A1,(A0)             ; *(long *)(intout+4)=picbuffer;
X          bra       selfok
X
X; This is the normal clipping procedure
X; ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
X
Xnormal:    move.l    D1,A0
X          move.l    8(A0),A0
X          move.w    (A0)+,clipx0        ; clipx0 = intin[0];
X          move.w    (A0)+,clipy0        ; clipy0 = intin[1];
X          move.w    (A0)+,clipx1        ; clipx1 = intin[2];
X          move.w    (A0),clipy1         ; clipy1 = intin[3];
X
Xgemrest:   movem.l   (A7)+,D0-A3         ; restore registers
Xgemquit:   move.l    gemold,-(A7)        ; put old GEM vector on the stack
X          rts                           ; and jump to it...
X
X          data
X
Xnoudtext: dc.b      27,'E',13,10,10,"    ",27,'p'," Desktop Picture V1.1
"
X          dc.b      27,'q',13,10,"    ",27,'p'," Noud van Kruysbergen
",27,'q',13,10,10,0
Xerror1:   dc.b      "    **** Error: DESKTOP.PI3 ? ****",13,10,0
Xerror2:   dc.b      "     Warning: DESKTOP.PI3 < 32034",13,10,0
Xfile1:    dc.b      "AUTO\DESKTOP.PI3",0
Xfile2:    dc.b      "DESKTOP.PI3",0
X          even
X
X          bss
```

```
X
Xstyle:    ds.w      1
Xinterior: ds.w      1
Xmode:     ds.w      1
Xclipx0:   ds.w      1
Xclipy0:   ds.w      1
Xclipx1:   ds.w      1
Xclipy1:   ds.w      1
Xgemold:   ds.l      1
Xpicbuffer:ds.l      1                   ; 32000 bytes are reserved from here.
X
X          end
X
SHAR_EOF
chmod 0600 DESKTOP2.S || echo "restore of DESKTOP2.S fails"
exit 0
```

```
From -7005384537836214485
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,5c058ffbb675e177,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-10 08:22:51 PST
Path: utzoo!attcan!utgpu!jarvis.csri.toronto.edu!mailrus!uwm.edu!gem.mps.ohio-
state.edu!apple!sun-barr!newstop!sun!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i087: truchet -- Display random Truchet tilings on mono screens
Keywords: C
Message-ID: <34505@grapevine.uucp>
Date: 10 Oct 89 15:22:51 GMT
Sender: news@grapevine.EBay
Reply-To: BD05@DKAUNI2.Bitnet (Roland Waldi)
Lines: 121
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: BD05@DKAUNI2.Bitnet (Roland Waldi)
Posting-number: Volume 2, Issue 87
Archive-name: truchet

Here is the source to the truchet tiling demo, which I submitted to
the binaries group earlier.

/*-------------------------------------------------------------*
 *                                                             *
 * TRUCHET2.C: main()                                          *
 * display random Truchet tiles on a monochrome screen         *
 *                                                             *
 * compiler used: LASER C (Atari ST)                           *
 *                                                             *
 * The comments may be used to get a better view of the        *
 * program structure: type 'EXCLUDE /**' for the top level     *
 * structure, 'EXCLUDE /***' to reveal the next level etc.     *
```

```
 * where 'EXCLUDE' should be replaced by the appropriate       *
 * command of your favourite editor.                           *
 *                                                             *
 *-----Version 1.00-----(C) 1989 Roland Waldi------------------*/
                                                /*  */
#include <osbind.h>                             /*  */
                                                /*  globals */
int  *screen[2];                                /** globals */
int   tile[2][16] = {128,128,192,64,            /** globals ... */
                     96,48,28,-4089,            /*** globals */
                     7168,1536,768,256,         /*** globals */
                     384,128,128,128,           /*** globals */
                                                /*** globals */
                     128,128,384,256,           /*** globals */
                     768,1536,7168,-4089,       /*** globals */
                     28,48,96,64,               /*** globals */
                     192,128,128,128};          /*** globals */
/*****  ........X.......   128          ........X.......   128 */
/*****  ........X.......   128          ........X.......   128 */
/*****  ........XX......   192          .......XX.......   384 */
/*****  .........X......    64          ...... X........   256 */
/*****  .........XX.....    96          ......XX........   768 */
/*****  ..........XX ...    48          .....XX.........  1536 */
/*****  ............XXX..    28         ...XXX..........  7168 */
/*****  XXXX.........XXX -4089         XXXX.........XXX -4089 */
/*****  ...XXX...........  7168         ............XXX..    28 */
/*****  .....XX.........  1536         ..........XX....    48 */
/*****  ......XX........   768         .........XX.....    96 */
/*****  .......X........   256         .........X......    64 */
/*****  .......XX.......   384         ........XX......   192 */
/*****  ........X.......   128         ........X.......   128 */
/*****  ........X.......   128         ........X.......   128 */
/*****  ........X.......   128         ........X.......   128 */
                                                /*** */
long RINT = 7716383l;                           /** globals for Rn */
long RMULT = 331804469l;                        /** globals for Rn */
                                                /** */
main()                                          /*  main */
{                                               /*  main */
  int actual;                                   /** declarations */
  register int i, n;                            /** declarations */
  register int  *c;                             /** declarations */
  long temp;                                    /** declarations */
                                                /** */
  if (Getrez() != 2)  Pterm(1);                 /** check resolution */
  actual = 1;                                   /**  initialize ... */
  screen[0] = (int  *)Physbase();               /*** initialize */
  temp = Malloc(32256l);                        /*** initialize */
  if (!temp)       Pterm(2);                    /*** initialize */
  screen[1] = (int  *)((temp+256)&0xFFFF00l);   /*** initialize */
  a_hidemouse();                                /*** initialize */
  RINT += Tgettime();                           /*** initialize */
                                                /** */
```

```
  for (n=0; n<1000; n++)                        /**  init loop */
  {                                             /**  init loop */
    copytile(0,n,actual);                       /*** init loop */
  }                                             /**  init loop */
  Setscreen(-1l,screen[actual],-1);             /**  initialize ... */
  Vsync();                                      /*** initialize */
                                                /** */
  while (!Cconis())                             /**  main loop */
  {                                             /**  main loop */
    actual = !actual;                           /*** main loop */
    fwcopy(screen[actual],screen[!actual],      /*** main loop */
           32000l);                             /*** main loop */
    n = Rn(1000);                               /*** main loop */
    c = screen[actual]                          /*** main loop */
          + (n/40) * 640 + (n%40) + 320;        /*** main loop */
    i = (*c)>>12;                               /*** flip tile */
    copytile(i,n,actual);                       /*** flip tile */
    Setscreen(-1l,screen[actual],-1);           /*** main loop */
    Vsync();                                    /*** main loop */
  }                                             /**  main loop */
                                                /** */
  Setscreen(-1l,screen[0],-1);                  /**  terminate ... */
  Vsync();                                      /*** terminate */
  a_showmouse();                                /*** terminate */
  Mfree(temp);                                  /*** terminate */
  Pterm(0);                                     /*** terminate */
}                                               /*  end main */
                                                /*  */
copytile(i,n,actual)                            /*  copytile */
register int i, actual; int n;                  /*  copytile */
{                                               /*  copytile */
  register int j;                               /** declarations */
  register int *start;                          /** declarations */
                                                /** */
  start = screen[actual]                        /**  get address ... */
          + (n/40) * 640 + (n%40);              /*** get address */
  for (j=0; j<16; j++)                          /**  copy loop */
  {                                             /**  copy loop */
    *(start + j*40) = tile[i][j];               /*** copy loop */
  }                                             /**  copy loop */
                                                /** */
  return;                                       /** copytile */
}                                               /*  end copytile */
                                                /*  */
int Rn(n) int n;                                /*  Rn */
/* Random number generator, return pseudo random number in [0...n-1] */
{                                               /*  Rn */
  RINT = (RINT * RMULT) - 1;                    /** Rn */
  return (int)( (((RINT & 0x7FFFFFFFl) >> 16)   /** Rn */
                 * n) /  0x8000l );             /** Rn */
}                                               /*  end Rn */
```

From 96336358592777018
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,a60da6ae7441a292,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-10 14:21:12 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!cs.utexas.edu!ginosko!usc!apple!
sun-barr!newstop!sun!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i088: yaff -- Yet Another File Finder
Keywords: shar, C
Message-ID: <34512@grapevine.uucp>
Date: 10 Oct 89 21:21:12 GMT
Sender: news@grapevine.EBay
Reply-To: uunet!unido!infbs!tubsibr!hafer (Udo Hafermann)
Lines: 542
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: uunet!unido!infbs!tubsibr!hafer (Udo Hafermann)
Posting-number: Volume 2, Issue 88
Archive-name: yaff

Here is yet another version of an Atari ST file finder. This one is
a no-frills version meant to be used from a shell (gulam, msh etc.);
it will run from the desktop though. As the basic machinery was
there, I went a little option-crazy and included stuff like filesize
counting etc.

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      FF.C
#      FF.MAN
#      MAKEFILE
#
sed 's/^X//' << 'SHAR_EOF' > FF.C &&
X
X
X/*
X * Searches recursively for all directory entries matching a given
X * specification beginning at the directory specified.
X * See ff.man for details.
X *
X * For ST-TOS, MWC 3.05.
X * History: 17-SEP-87
X *          06-JUL-89 regexp as filespec; -x option
X *          24-JUL-89 file sizes
X */
X
X#include <stdio.h>
X#include <osbind.h>
X#include <bios.h>
```

```
X#include <ctype.h>
X
X/* requirements:  */
X
X#define PATHLEN       256  /* max. length of paths we can manage
     */
X                 /* (more than GEMDOS can, hopefully)  */
X#define BUFLEN        1000 /* string buffer size
     */
X#define  CLUSTERSIZE   1024 /* cluster size assumed for -k
          */
X#define  STACKSIZE 10000    /* recursive search might need it
     */
X
X/* file types:    */
X
X#define FWPROT    (0x01)        /* write protected
     */
X#define FHIDE     (0x02)        /* hidden              */
X#define FSYS (0x04)        /* system file
     */
X#define FVOL (0x08)        /* volume label
     */
X#define FSUB (0x10)        /* subdirectory
     */
X#define FARCH     (0x20)        /* archive bit
          */
X
X/* return codes:  */
X
X#define E_OK (0L)      /* no error                 */
X#define E_FILNF   (-33L)        /* file not found
     */
X#define E_PTHNF   (-34L)        /* path not found
     */
X#define E_NMFIL   (-49L)        /* no more files
     */
X
X/* check macro for "." and ".." file entries:
     */
X#define NOTDOT(fname) (fname[0]!='.' || (fname[1]!='\0' && fname[1]!='.'))
X
X/* stack request: */
Xlong    _stksize = STACKSIZE;
X
X/* variables:     */
X
Xstatic   char *whoami = "ff";    /* if non-empty, will point to argv[0]
     */
Xstatic   char *olddta;  /* dta backup (do we really need that?)
     */
X
Xstatic   char filespec[PATHLEN];
```

```
X                 /* wildcard-spec of filenames        */
X                 /* to be searched for           */
Xstatic   char path[PATHLEN],     /* path of directory
         */
X                 /* currently being searched     */
X         drivespec[BUFSIZ],
X                 /* drive specification part of filespec
     */
X         buf1[BUFSIZ],
X         *xflag;       /* pointer to "-x cmd", if not null
     */
X
Xstatic   int  cflag,        /* if true, -c flag
     */
X         qflag,        /* if true, -q flag             */
X         vflag,        /* ... and so on ...            */
X         hflag,
X         sflag,
X         dflag,
X         tflag,
X         kflag,
X         count,        /* # matches so far             */
X         ftypes;       /* attribute bits           */
X
Xstatic   long total;        /* accumulated size
     */
X
Xunsigned char     fattset, /* if non-zero, these atts must be set
     */
X         fattunset = FSUB;
X                 /* these atts must be unset     */
X
Xstatic terminate(status)
Xint status;
X{
X    /* terminate program returning status*/
X    Fsetdta(olddta);   /* is this necessary?           */
X    if (hflag) {
X         fprintf(stderr, "Type <Return>: ");
X         getchar();
X    }
X    exit(status);
X}
X
Xstatic info(full)
Xint full;
X{
X    /* usage info, short or full     */
X    fprintf((full ? stdout : stderr),
X       "Usage:    %s [-?] [-hcqvsktd] [-aatts] [-iatts] [-x command]
'filespec'\n",
X       whoami);
X    if (full) {
```

```
X       printf("\tPrints all filenames matching 'filespec', recursively
searching\n");
X       printf("\tsubdirectories from the one specified (default: current).\n");
X       printf("\tWildcards:  *  ?  []\n");
X       printf("Flags:\n");
X       printf("\t-h\tWait before terminating\n");
X       printf("\t-c\tPrint match count\n");
X       printf("\t-q\tQuiet: print no filenames\n");
X       printf("\t-v\tVerbose: list directories searched (on stderr)\n");
X       printf("\t-s\tList file sizes\n");
X       printf("\t-d\tList subtotals for subdirectories\n");
X       printf("\t-t\tSuppress total size\n");
X       printf("\t-k\tList file sizes, rounded to multiples of %d\n",
X             CLUSTERSIZE);
X       printf("\t-aatts\tOnly list files with these attributes\n");
X       printf("\t-iatts\tIgnore files with these attributes (default: d)\n");
X       printf("\t-x cmd\tExecute command, where %%%% is replaced by filename\n");
X       printf("Exit status: The match count\n");
X    }
X    terminate (-1);
X}
X
Xstatic unsigned char parseatts (cp)
Xchar *cp;
X{
X    /* cp points to string of letters r,h,s,d, or m.
X       Returns bitmap of attributes.              */
X    unsigned char atts = 0;
X    char c;
X
X    while (c = *cp++) {
X         if (c=='r')
X              atts |= FWPROT;
X         else if (c=='h')
X              atts |= FHIDE;
X         else if (c=='s')
X              atts |= FSYS;
X         else if (c=='v')
X              atts |= FVOL;
X         else if (c=='d')
X              atts |= FSUB;
X         else if (c=='m')
X              atts |= FARCH;
X    }
X    return atts;
X}
X
Xstatic void str_tolower(s)
Xchar    *s;
X{
X    /* convert s to lower case  */
X    while (*s) {
X         (*s) = tolower(*s);
```

```
X         s++;
X    }
X}
X
Xstatic void str_repl (target, source, s1, s2)
Xchar    *target, *source, *s1, *s2;
X{
X    /* copies source into target, replacing each occurrence
X       of s1 with s2                          */
X    unsigned int s1len = strlen(s1), s2len = strlen(s2);
X
X    while (*source) {
X         if (0 == strncmp(source, s1, s1len)) {
X              /* substitute s2 for s1:    */
X              strcpy (target, s2);
X              target += s2len;
X              source += s1len;
X         } else {
X              *target++ = *source++;
X         }
X    }
X    *target = '\0';
X}
X
Xmain(argc, argv)
Xint argc;
Xchar *argv[];
X{
X    olddta = (char *)Fgetdta(); /* restored by terminate()
     */
X
X    /* First, save program name for messages:          */
X    if (argv[0][0])
X         whoami = argv[0];
X
X    /* Process options:                              */
X    while (--argc) {
X         char c,
X              *cp;
X
X         cp = *++argv;
X         if (cp[0] != '-') {
X              /* not an option: must be first filespec  */
X              break;
X         }
X         if (strlen(cp) < 2) {
X              /* invalid option               */
X              info(0);
X         }
X         /* process these options:                */
X         str_tolower(cp);
X         while (c = *++cp) {
X              if (c == 'c')
```

```
X                 cflag = -1;
X            else if (c=='h')
X                 hflag = -1;
X            else if (c=='q')
X                 qflag = -1;
X            else if (c=='v')
X                 vflag = -1;
X            else if (c=='s')
X                 sflag = -1;
X            else if (c=='t')
X                 tflag = -1;
X            else if (c=='k')
X                 kflag = sflag = -1;
X            else if (c=='d')
X                 dflag = -1;
X            else if (c=='x') {
X                 xflag = *++argv;
X                 argc--;
X            } else if (c=='a') {
X                 /* set attributes           */
X                 fattset = parseatts (++cp);
X                 break;
X            } else if (c=='i') {
X                 /* unset atts               */
X                 fattunset = 0;
X                 fattunset = parseatts (++cp);
X                 break;
X            } else {
X                 /* unknown option (possibly '?') */
X                 info(c=='?');
X            }
X        }
X    }
X
X    /* set up file types for fsnext:               */
X    ftypes = FSUB | FHIDE | FWPROT | FSYS | FARCH;
X
X    /* apply principle of least surprise (we ignore subdirs by
X             default, remember):                   */
X    if (fattset & FSUB)
X         fattunset &= ~FSUB;
X
X    /* now process the filespecs:                      */
X    if (!argc) {
X         info(0); /* we must have at least one...      */
X    }
X
X    while (argc--) {
X         char *cp, *dp, *ep;         /* quite a lot, eh?
     */
X                              /* but wait and see... */
X         int  have_drive;
X
```

```
X        /* parse the argument into:
X           filespec - argument for pnmatch
X           drivespec -     drive portion (possibly empty)
X           path  -    absolute part of path (first subdir to search)
X        */
X
X        drivespec[0] = 0;
X        cp = *argv;   /* argument pointer             */
X        dp = path;    /* path  construction pointer
     */
X        ep = path;    /* current end of absolute path
     */
X        have_drive = 0;    /* true when drive spec is passed
     */
X        do {
X             *dp++ = tolower(*cp);
X             if (*cp == '\\') {
X                  /* drivespec is complete:   */
X                  have_drive = 1;
X                  /* mark position:  */
X                  ep = dp;
X             } else if (*cp == ':' && !have_drive) {
X                  /* end of real drive spec!  */
X                  /* copy it:   */
X                  *(dp-1) = 0;
X                  strcpy (drivespec, path);
X                  have_drive = 1;
X                  /* intialize path appropriately: */
X                  dp = path;
X                  *dp++ = 'x';  *dp++ = ':';
X                  ep = dp;
X             }
X        } while (*cp++);
X        /* copy end of path to filespec: */
X        strcpy (filespec, ep);
X        /* and terminate path after last backslash:    */
X        *ep = 0;
X
X        if (drivespec[0]) {
X             /* drive specified; we must try all: */
X             char drive;
X             long drivemap;
X             char *hugo = "x";
X
X             drivemap = Drvmap();
X             for (drive='a'; drive<='p'; drive++) {
X                  if (drivemap & 0x1) {
X                       path[0] = hugo[0] = drive;
X                       if (pnmatch(hugo, drivespec, 0))
X                            recursive(*argv);
X                  }
X                  drivemap >>= 1;
X             }
```

```
X         } else {
X              recursive(*argv);
X         }
X         argv++;
X    }
X
X    /* Report results:                          */
X    if (cflag) {
X         if (vflag)
X              printf ("Entries: ");
X         printf ("%d\n", count);
X    } else {
X         if (vflag && !count)
X              fprintf (stderr, "%s: no match.\n", whoami);
X    }
X    if (sflag && !(tflag || dflag))
X         if (vflag)
X              printf ("Total size: %ld\n", total);
X         else
X              printf ("%10ld\n", total);
X
X    terminate(count);
X}
X
Xstatic recursive (arg)
Xchar    *arg;    /* the original search argument (for verbose mode)
     */
X{
X    /* search for files matching filespec in current path and subdirs */
X    struct Buff {
X         unsigned char reserved[21];
X         unsigned char type;
X         unsigned short    time;
X         unsigned short    date;
X         unsigned long size;
X         unsigned char name[14];
X         } buffer;/* dta buffer for this path     */
X    int  pp,      /* path marker for backtracking      */
X         status;       /* GEMDOS return code           */
X    long oldtotal = total;
X                  /* so that we can work out subtotal  */
X
X    /* first, search current dir:                       */
X    if (vflag)
X         fprintf (stderr,
X              "%s: Searching for '%s' in '%s'\n", whoami, arg, path);
X    Fsetdta(&buffer);
X    pp = strlen(path);
X    strcat (path, "*.*");
X    status = Fsfirst(path, ftypes);
X    while (status == E_OK) {
X         /* does this entry match?                */
X         str_tolower(buffer.name);
```

```
X        if (pnmatch(buffer.name, filespec, 0)
X            && (!fattset || fattset == (fattset & buffer.type))
X            && (! (fattunset & buffer.type))
X            && NOTDOT (buffer.name)) {
X             path[pp] = '\0';
X             strcat(path, buffer.name);
X             if (sflag || dflag) {
X                  long s = buffer.size;
X                  if (kflag)
X                       s = CLUSTERSIZE *
X                       ((s+CLUSTERSIZE-1)/CLUSTERSIZE);
X                  if (!qflag)
X                       printf("%10ld   ", s);
X                  total += s;
X             }
X             if (!qflag) {
X                  printf("%s\n", path);
X             }
X             count++;
X             if (xflag) {
X                  str_repl (buf1, xflag, "%%", path);
X                  system (buf1);
X             }
X             path[pp] = '\0';
X             strcat(path, "*.*");
X        }
X        status = Fsnext();
X   }
X   path[pp] = '\0';
X   /* check return code for unusual stuff:   */
X   if (status != E_FILNF && status != E_NMFIL) {
X        /* unexpected return code!           */
X        fprintf (stderr,
X             "%s '%s': Oops - unexpected error code: %d\n",
X             whoami, path, status);
X        terminate(status);
X   }
X   /* Now search all subdirs:                    */
X   strcat(path, "*.*");
X   status = Fsfirst(path, FSUB|FHIDE|FSYS|FWPROT|FARCH);
X   path[pp] = '\0';
X   while (status==E_OK) {
X        if (buffer.type&FSUB  &&  NOTDOT(buffer.name)) {
X             str_tolower(buffer.name);
X             strcat (path, buffer.name);
X             strcat (path, "\\");
X             recursive(arg);
X             path[pp] = '\0';
X             Fsetdta(&buffer);
X        }
X        status = Fsnext();
X   }
X   if (dflag)
```

```
X        printf ("%10ld %s\n", total-oldtotal, path);
X}
SHAR_EOF
chmod 0600 FF.C || echo "restore of FF.C fails"
sed 's/^X//' << 'SHAR_EOF' > FF.MAN &&
X
XFF (1)           ATARI ST UTILITIES              FF (1)
X
X
XNAME
X    ff  -  _f_i_l_e_f_i_n_d: recursive search for files
X
XSYNOPSIS
X
X    ff [-cqvsktd] [-aatts] [-iatts] [-x command] 'filespec'
X
XDESCRIPTION
X
X    ff takes everything up to the last backslash as the directory to
X    start searching (default is the current directory).  The rest of
X    filespec is the pattern.  ff will list all files (with their complete
X    paths) whose last names match this pattern, recursively scanning all
X    subdirectories.  filespec may begin with a drive specification
X    (ending with a colon; a backslash should follow), which is matched
X    against all drives present.
X
X    ff uses the MWC routine pnmatch() for the pattern match;  thus,
X    permissible wildcards are * (matches any sequence of characters),
X    ? (matches any one character), and character lists enclosed in
X    brackets ([]).
X
X    Flags:
X         -c   _C_o_u_n_t: print number of files found.
X         -q   _Q_u_i_e_t: print no filenames or extra messages.
X         -v   _V_e_r_b_o_s_e: list directories searched.
X         -aatts   Include only files with these _a_t_t_r_i_b_u_t_e_s
X              (r,h,s,d,m for readonly, hidden, system, subdir,
X              modified flag, respectively).
X         -iatts   _I_g_n_o_r_e files with these attributes (default: d).
X         -s   _S_i_z_e: list file sizes and total.
X         -t   Suppress _t_o_t_a_l.
X         -k   _k_B_y_t_e: round sizes to multiples of 1024 (implies -s).
X              On systems with a cluster size of 1 kByte, this should
X              report the space actually used by the files.
X         -d   List subtotal for each _s_u_b_d_i_r_e_c_t_o_r_y (implies -s).
X         -x cmd   For each match, _e_x_e_c_u_t_e command cmd
X              using msh, where %% is replaced with the filename.
X
X    Exit status:  The number of matching filenames found, or a negative
X              error number.
X
XEXAMPLES
X
```

```
X    ff 'hugo' find files named hugo, beginning at the current
X              directory.
X    ff -c '*' find all files, beginning at the current directory,
X              reporting the number of files found.
X    ff '?:\*.c'   find all C source files on all existing drives.
X    ff -am '[cd]:\hugo\*'  find all files, beginning at the
X              subdirectories "c:\hugo" and "d:\hugo", which have
X              the "modified" attribute set.
X
XRESTRICTIONS
X
X    - For the -x option, msh.prg must be in the search path.
X    - The files are listed in the order in which they appear in the
X      directory (i.e., not sorted in any way).
X    - As the backslash is the directory seperator, there is no escape
X      character available for the wildcard characters.
X
XNOTES
X
X    - The filespec must be enclosed in quotes if the shell used
X      would otherwise expand the wildcards.  Contemplate the
X      difference between "ff *.c" and "ff '*.c'" under gulam (or msh).
X    - Arguments are forced to lower case (for use from the desktop under
X      old TOS versions).
X    - "ff -qkd 'dir\*'" is roughly equivalent to "du dir".
X    - Very convenient under msh using backquotes:
X      rm `ff '*'`  is equivalent to UNIX rm -r, with the
X      exception that subdirectories are not deleted, and that the msh
X      cannot process an unlimited number of arguments that way.
X
XWARNINGS
X
X    As ff may traverse quite a few directories, you might run into
X    the _4_0_-_f_o_l_d_e_r_-_b_u_g (discussed at length elsewhere).
X
XTODO
X    - There should be support for matching against the file timestamps.
X    - ff should, instead of just calling system(), use the gulam shell
X      interface if present.
X    - There should be a replacement for the pnmatch() escape character.
X
XFILES
X    ff.ttp
X
XAUTHOR
X
X    hafer@tubsibr july 1989
X
SHAR_EOF
chmod 0600 FF.MAN || echo "restore of FF.MAN fails"
sed 's/^X//' << 'SHAR_EOF' > MAKEFILE &&
X
Xff.ttp: ff.c
```

```
X    cc -s -o ff.ttp ff.c
X
X
SHAR_EOF
chmod 0600 MAKEFILE || echo "restore of MAKEFILE fails"
exit 0


From 616855982150382343
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,1321b32def43ce51,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-10 14:23:08 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!wuarchive!gem.mps.ohio-
state.edu!apple!sun-barr!newstop!sun!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i089: strings -- Find text strings in files
Keywords: shar, C
Message-ID: <34513@grapevine.uucp>
Date: 10 Oct 89 21:23:08 GMT
Sender: news@grapevine.EBay
Reply-To: uunet!unido!infbs!tubsibr!hafer (Udo Hafermann)
Lines: 378
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: uunet!unido!infbs!tubsibr!hafer (Udo Hafermann)
Posting-number: Volume 2, Issue 89
Archive-name: strings

[This arrived packaged with the "yaff" program. -sg]

#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      DEBUG.H
#      GETOPT.C
#      GETOPT.H
#      MAKEFILE
#      STRINGS.C
#      STRINGS.MAN
#
sed 's/^X//' << 'SHAR_EOF' > DEBUG.H &&
X/*
X * debug.h - debugging macros
X *
X */
X
X/*
X * dd(value, format) creates a printf-statement which dumps
X * the given value in the specified format.
X */
```

```
X
X#if NDEBUG
X#define  DD(value, format)
X#else
X#define  DD(value, format)  printf("[" __FILE__ "(%d): " #value "=" #format "]\n",
__LINE__, value);
X#endif
X
X/* End of debug.h */
SHAR_EOF
chmod 0600 DEBUG.H || echo "restore of DEBUG.H fails"
sed 's/^X//' << 'SHAR_EOF' > GETOPT.C &&
X/*
X**  Return options and their values from the command line.
X**
X**  This comes from the AT&T public-domain getopt published in mod.sources.
X*/
X
X/*  adapted for MWC by hafer@infbs
X*/
X
X#include <stdio.h>
X
X#define TYPE int
X
X#define ERR(s, c) if(opterr){fprintf(stderr, "%s%s%c\n", argv[0], s, c);}
X
Xextern int strcmp();
Xextern char *index();
X
X/* external vars */
X
Xint opterr = 1;   /* if reset to 0, getopt is silent on errors
     */
Xint optind = 1;   /* argument number of current option
     */
Xint optopt;       /* current option character          */
Xchar    *optarg; /* pointer to extra argument of current option
     */
X
Xint
Xgetopt(argc, argv, opts)
Xint argc;
Xchar    **argv, *opts;
X{
X    static int sp = 1;
X    register int c;
X    register char *cp;
X
X    if(sp == 1)
X         if(optind >= argc ||
X            argv[optind][0] != '-' || argv[optind][1] == '\0')
X              return(EOF);
```

```
X         else if(strcmp(argv[optind], "--") == 0) {
X              optind++;
X
X         }
X    optopt = c = argv[optind][sp];
X    if(c == ':' || (cp=index(opts, c)) == NULL) {
X         ERR(": illegal option -- ", c)
X         if(argv[optind][++sp] == '\0') {
X              optind++;
X              sp = 1;
X         }
X         return('?');
X    }
X    if(*++cp == ':') {
X         if(argv[optind][sp+1] != '\0')
X              optarg = &argv[optind++][sp+1];
X         else if(++optind >= argc) {
X              ERR(": option requires an argument -- ", c)
X              sp = 1;
X              return('?');
X         } else
X              optarg = argv[optind++];
X         sp = 1;
X    } else {
X         if(argv[optind][++sp] == '\0') {
X              sp = 1;
X              optind++;
X         }
X         optarg = NULL;
X    }
X    return(c);
X}
X
X
SHAR_EOF
chmod 0600 GETOPT.C || echo "restore of GETOPT.C fails"
sed 's/^X//' << 'SHAR_EOF' > GETOPT.H &&
X
X/*
X * getopt.h -     declarations for using getopt (needs <stdio.h>)
X */
X
Xextern  int  opterr;  /* if reset to 0, getopt is silent on errors
     */
Xextern  int  optind;  /* argument number following current option
     */
Xextern  int  optopt;  /* current option character
     */
Xextern  char *optarg;/* pointer to extra argument of current option
     */
X
Xextern  int  getopt(/* int argc, char **argv, char *opts */);
X    /* for each call, returns next option letter or EOF. */
```

```
X    /* opts is a list of permissible option letters, where a letter
X       may be followed by a colon to indicate an expected parameter */
SHAR_EOF
chmod 0600 GETOPT.H || echo "restore of GETOPT.H fails"
sed 's/^X//' << 'SHAR_EOF' > MAKEFILE &&
X
XPROGRAM= strings.ttp
XOBJ= strings.o getopt.o
X
XCFLAGS=-DNDEBUG
XLDFLAGS = -s
X
X$(PROGRAM):  $(OBJ)
X    cc -o $(PROGRAM) $(OBJ)
X
Xstrings.o:   strings.c debug.h getopt.h
X
Xgetopt.o:    getopt.c getopt.h
X
X
X
SHAR_EOF
chmod 0600 MAKEFILE || echo "restore of MAKEFILE fails"
sed 's/^X//' << 'SHAR_EOF' > STRINGS.C &&
X
X/*
X * strings.c -   extract strings from binary file
X * hafer@infbs   06-AUG-89
X */
X
X#include <stdio.h>
X#include "debug.h"
X#include "getopt.h"
X
Xextern  char *index();
X
X#define BUFLEN        1024
X#define MINDEFAULT    4
X
Xtypedef struct {
X    int  magic;
X    long text;
X    long data;
X    long bss;
X    long symtab;
X    long reserved1, reserved2;
X    int  flags;
X} header;
X
Xstatic  char buffer[BUFLEN];
Xstatic  int  bufcnt,
X        aflag,
X        lflag,
```

```
X         pflag,
X         nflag,
X         vflag,
X         minlen = MINDEFAULT;
Xstatic   char *options = "adoxlnuvm:c:",
X         *whoami = "strings",
X         *format = "%8ld ",
X         extrachars[BUFLEN],
X         *umlaute = "\204\224\201\216\231\232\236",     /* "" */
X         *filename;
Xstatic   long pos,
X         length,
X         offset;
X
Xstatic   void addchar(ch)
Xint ch;
X{
X    /* adds a char to buffer */
X    if (bufcnt < BUFLEN-1) {
X         buffer[bufcnt++] = ch;
X    }
X}
X
Xstatic   void printbuf()
X{
X    /* prints buffer */
X    buffer[bufcnt] = '\0';
X    if (bufcnt >= minlen) {
X         if (pflag)
X              printf (format, pos+offset);
X         if (lflag)
X              printf ("%s\n", buffer);
X         else
X              printf ("\"%s\"\n", buffer);
X    }
X    bufcnt = 0;
X}
X
Xstatic   void usage()
X{
X    fprintf (stderr, "Usage: %s [-%s] <filename>\n", whoami, options);
X    exit (-1);
X}
X
Xmain (argc, argv)
Xint argc;
Xchar     **argv;
X{
X    int  ch;
X    FILE *fp;
X    header   h;
X
X    if (*argv[0])
```

```
X         whoami = argv[0];
X    opterr = 0;
X    while (EOF != getopt(argc, argv, options)) {
X         if (optopt == 'a') {
X              aflag = -1;
X         } else if (optopt == 'l') {
X              lflag = -1;
X         } else if (optopt == 'x') {
X              pflag = -1;
X              format = "%8lx ";
X         } else if (optopt == 'o') {
X              pflag = -1;
X              format = "%8lo ";
X         } else if (optopt == 'd') {
X              pflag = -1;
X         } else if (optopt == 'n') {
X              nflag = -1;
X         } else if (optopt == 'v') {
X              vflag = -1;
X         } else if (optopt == 'c') {
X              strcat (extrachars, optarg);
X         } else if (optopt == 'u') {
X              strcat (extrachars, umlaute);
X         } else if (optopt == 'm') {
X              minlen = atoi (optarg);
X              if (minlen == 0)
X                   minlen = MINDEFAULT;
X         } else {
X              usage();
X         }
X    }
X    /* check whether there is exactly one argument left:
     */
X    if (argc-optind != 1) {
X         usage();
X    }
X    filename = argv[optind];
X    if (!(fp = fopen(filename, "rb"))) {
X         fprintf (stderr, "%s: cannot open file '%s'\n",
X              whoami, filename);
X         exit (-33);
X    }
X
X    if (aflag) {
X         offset = 0;
X         length = -1;
X    } else {
X         if (!fread(&h, sizeof(h), 1, fp) || h.magic != 0x601a) {
X              fprintf (stderr, "%s: unknown format: '%s'\n",
X                   whoami, filename);
X                   exit (-1);
X         }
X         if (vflag) {
```

```
X             printf ("    text     data      bss   symtab\n");
X             printf (format, h.text);
X             printf (format, h.data);
X             printf (format, h.bss);
X             printf (format, h.symtab);
X             printf ("\n");
X        }
X        offset = sizeof(h) + h.text;
X        length = h.data;
XDD(offset, %ld)
XDD(length, %ld)
X        fseek (fp, offset, 0);
X    }
X
X    /* off we go: */
X    for (pos; pos != length; pos++) {
X         ch = getc(fp);
X         if (feof(fp)) {
X              /* this shouldn't happen if the header is valid... */
X              fclose (fp);
X              exit(0);
X         } else if (ch == '\0' ||
X                   (nflag && (ch == '\n' || ch == '\r'))) {
X              printbuf();
X         } else if (ch == '"' || ch == '\\') {
X              if (lflag) {
X                   addchar(ch);
X              } else {
X                   addchar('\\');
X                   addchar(ch);
X              }
X         } else if (ch>=' ' && ch<127) {
X              addchar(ch);
X         } else if (ch=='\n' || ch=='\t') {
X              if (lflag) {
X                   addchar(ch);
X              } else {
X                   addchar('\\');
X                   addchar((ch=='\n' ? 'n' : 't'));
X              }
X         } else if (*extrachars && index(extrachars, ch)) {
X              addchar(ch);
X         } else {
X              bufcnt = 0;
X         }
X    }
X    exit(0);
X}
SHAR_EOF
chmod 0600 STRINGS.C || echo "restore of STRINGS.C fails"
sed 's/^X//' << 'SHAR_EOF' > STRINGS.MAN &&
XSTRINGS(1)            USER COMMANDS
X
```

```
X
XNAME
X    strings  -  extract strings from binary file
X
XSYNOPSIS
X    strings [-adoxnv] [-m <number>] [-c chars] filename
X
XDESCRIPTION
X    Extracts and prints sequences of printable characters, terminated by
X    a _z_e_r_o _b_y_t_e, from the _d_a_t_a segment of the specified file, which
is
X    assumed to be a GEMDOS executable.
X    Strings are listed in C-format, i.e. as they would be accepted
X    as string constants in C.  Strings shorter than 4 characters are
X    ignored.  Printable characters are '\t', '\n', and ASCII 32 to 126.
X
X    Options:
X
X    -a   All:  Search the _c_o_m_p_l_e_t_e file (regardless of its format).
X    -d   _D_e_c_i_m_a_l: Precede each string with its _p_o_s_i_t_i_o_n in the
file.
X    -o   _O_c_t_a_l: Same as -d, only in octal.
X    -x   _H_e_x_a_d_e_c_i_m_a_l: Same as -d, only in hex.
X    -l   Literal: Print strings _l_i_t_e_r_a_l_l_y (i.e., not in C-format).
X    -n   Newline: Accept '\n' and '\r' as string _t_e_r_m_i_n_a_t_o_r_s.
X    -m <no>  _M_i_n_i_m_u_m length:  Ignore strings shorter than <no> characters.
X    -c <ch>  _C_h_a_r_a_c_t_e_r_s: Regard the specified chars as
_p_r_i_n_t_a_b_l_e.
X    -u   _U_m_l_a_u_t_s: Regard "umlaut" characters as printable ("").
X    -v   _V_e_r_b_o_s_e:  List the segment sizes.
X
XRESTRICTIONS
X    Printing of strings is limited to the internal buffer size of 1024
X    characters.
X
XBUGS
X    When printing in C-format, characters greater than 126 are not
X    escaped by '\\ddd' (only with -c or -u options).
X
XAUTHOR
X    hafer@infbs.uucp   August 1989
X
SHAR_EOF
chmod 0600 STRINGS.MAN || echo "restore of STRINGS.MAN fails"
exit 0


From -8915872072759063850
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,b21635cce54a7a08,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-12 13:41:01 PST
Path: utzoo!attcan!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!
```

gem.mps.ohio-state.edu!apple!sun-barr!newstop!sun!grapevine!
koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i090: curses -- Screen updating package part01/02
Keywords: shar, C
Message-ID: <34528@grapevine.uucp>
Date: 12 Oct 89 20:41:01 GMT
Sender: news@grapevine.EBay
Reply-To: gray@hplb.hpl.hp.com (Graham Higgins)
Lines: 1633
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: gray@hplb.hpl.hp.com (Graham Higgins)
Posting-number: Volume 2, Issue 90
Archive-name: curses/part01

Following a request on comp.sys.atari.st, here's the source to
curses. Apologies to the original author, TWINKLE.TOS didn't
appear in the posting where I obtained this, the poster wanted to
post plain text - understandable. [Since that's the way I want
sources submitted, VERY understandable. :) -sg]

Cheers,

Graham
======

Graham Higgins | Phone: (0272) 799910 x 24060
Hewlett-Packard Labs | gray@hpl.hp.co.uk
Bristol | gray%hplb.uucp@ukc.ac.uk
U.K. | gray@hplb.hpl.hp.com

Disclaimer: My opinions above are exactly that, mine and opinions.

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
# This is part 1 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
#
#    Run the following text with /bin/sh to create:
#      curses.c
#      curses.doc
#      curses.h
#      curses.man
#      portab.h
#      readme.too
#      readme.txt
#      twinkle.c
#
```

```
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
sed 's/^X//' << 'SHAR_EOF' > curses.c &&
X/* #[curses: simple-minded implementation of the curses package
                    */
X/* #[intro: introduction
                              */
X/*
                                   */
X/* This source and resulting object may be modified, used or distributed by */
X/* anyone who so desires under the following conditions :                    */
X/*
                                   */
X/*      1) This notice and the copyright notice shall not be removed or     */
X/*         changed.
                                   */
X/*      2) No credit shall be taken for this source and resulting objects   */
X/*      3) This source or resulting objects is not to be traded, sold or    */
X/*         used for personal gain or profit.
                         */
X/*      4) Inclusion of this source or resulting objects in commercially    */
X/*         available packages is forbidden without written consent of the   */
X/*         author of this source.
                              */
X/*                                                                          */
X/* #]intro:
                                   */
X/* #[include: include files
                                   */
X#include "portab.h"
X#include "osbind.h"
X#include "curses.h"
X/* #]include:
                                   */
X/* define supervisor mode entry forgotten in osbind.h
                         */
X#define Supexec(a)    xbios(38,a)
X/* terminal states : cooked, cbreak and raw
                              */
X#define COOKED 0
X#define CBREAK 1
X#define RAW    2
X/* define address of system-variable _conterm for kbshift return value
         */
X#define CONTERM       (*((char *)0x484))
X/* #[globals: definition of global variables
                         */
X/* #[curglob: define the following variables as external in curses.h
         */
XWINDOW *curscr ;
XWINDOW *stdscr ;
```

```
Xchar    *Def_term = "AtariST" ;
Xbool    My_term = FALSE ;
Xchar    *ttytype = "Atari ST-series computer, VT52/CURSES mode" ;
Xint     LINES = 25 ;
Xint     COLS = 80 ;
Xint     ERR = 0 ;
Xint     OK = 1 ;
X/* #]curglob:
                                  */
Xbool   _doecho = FALSE ;
Xbool   _nlmap = FALSE ;
Xbool   _curini = FALSE ;
XBYTE    _modinp = 0 ; /* set input to Un*x compatability, i.e only char's
    */
XBYTE    _csry = 0 ;
XBYTE    _csrx = 0 ;
Xchar *cpyrid = "Copyright, R. van't Veen. All rights reserved. 1987" ;
X
Xoverlay "curses"  /* put in separate segment */
X
X/* #]globals:
                                  */
X/* #[miscel: miscellaneous curses functions
                                */
X/* #[initscr: initialize the curses package
                                */
Xinitscr()
X{
X    WORD i,j ;
X    WINDOW *newwin() ;
X
X    if ( _curini )
X         {
X         wrefresh(curscr) ;
X         if ( cpyrid == NULL )
X              Pterm(-12L) ;
X         ERR = 1 ;
X         OK = 0 ;
X         return ;
X         } ;
X    _curini = 1 ;
X    curscr = newwin(0,0,0,0) ;
X    if ( curscr == NULL )
X         {
X         ERR = 1 ;
X         OK = 0 ;
X         return ;
X         } ;
X    stdscr = newwin(0,0,0,0) ;
X    if ( stdscr == NULL )
X         {
X         ERR = 1 ;
X         OK = 0 ;
```

```
X        return ;
X        } ;
X    Bconout(2,'\033') ;
X    Bconout(2,'E') ;
X    Bconout(2,'\033') ;
X    Bconout(2,'w') ;
X    Bconout(2,'\033') ;
X    Bconout(2,'f') ;
X    _csry = 0 ;
X    _csrx = 0 ;
X    ERR = 0 ;
X    OK = 1 ;
X}
X/* #]initscr:
                                  */
X/* #[endwin: end of curses package
                                */
Xendwin()
X{
X    if ( !_curini )
X        {
X        ERR = 1 ;
X        OK = 0 ;
X        return ;
X        } ;
X    _curini = 0 ;
X    delwin(stdscr) ;
X    delwin(curscr) ;
X    _movcur(LINES,0) ;
X    Bconout(2,'\033') ;
X    Bconout(2,'e') ;
X    ERR = 1 ;
X    OK = 0 ;
X}
X/* #]endwin:
                                */
X/* #[newwin: create a new window for the user
                           */
XWINDOW *newwin(l,c,by,bx)
Xint l,c,bx,by ;
X{
X    WINDOW *tmp ;
X    WORD   i, j, nl, nc ;
X    char *malloc() ;
X
X    tmp = ( WINDOW * ) malloc(sizeof(WINDOW)) ;
X    if ( tmp == NULL )
X        {
X        ERR = 1 ;
X        OK = 0 ;
X        return(NULL) ;
X        } ;
X    tmp->_curx = 0 ;
```

```
X    tmp->_cury = 0 ;
X    tmp->_clear = 0 ;
X    tmp->_leave = 0 ;
X    tmp->_scroll = 0 ;
X    if ( l == 0 )
X         nl = LINES - by ;
X    else
X         nl = l ;
X    if ( c == 0 )
X         nc = COLS - bx ;
X    else
X         nc = c ;
X    if ( nl < 1 || nl > LINES || nc < 1 || nc > COLS )
X         {
X         free(tmp) ;
X         ERR = 1 ;
X         OK = 0 ;
X         return(NULL) ;
X         } ;
X    tmp->_maxy = nl - 1 ;
X    tmp->_maxx = nc - 1 ;
X    if ( nl == LINES && nc == COLS )
X         tmp->_flags = _FULLWIN ;
X    else
X         tmp->_flags = 0 ;
X    if ( by < 0 || by >= LINES || bx < 0 || bx >= COLS )
X         {
X         free(tmp) ;
X         ERR = 1 ;
X         OK = 0 ;
X         return(NULL) ;
X         } ;
X    tmp->_begy = by ;
X    tmp->_begx = bx ;
X    tmp->_y = ( WORD ** ) malloc(sizeof(WORD *)*(tmp->_maxy+1)) ;
X    if ( tmp->_y == NULL )
X         {
X         free(tmp) ;
X         ERR = 1 ;
X         OK = 0 ;
X         return(NULL) ;
X         } ;
X    tmp->_y[0] = ( WORD * ) malloc(sizeof(WORD)*(tmp->_maxy+1)*(tmp->_maxx+1)) ;
X    if ( tmp->_y[0] == NULL )
X         {
X         free(tmp->_y) ;
X         free(tmp) ;
X         ERR = 1 ;
X         OK = 0 ;
X         return(NULL) ;
X         } ;
X    for ( i = 1 ; i <= tmp->_maxy ; i++ )
X         tmp->_y[i] = tmp->_y[0] + i * ( tmp->_maxx + 1 ) ;
```

```
X    for ( i = 0 ; i <= tmp->_maxy ; i++ )
X         {
X         for ( j = 0 ; j <= tmp->_maxx ; j++ )
X              tmp->_y[i][j] = ' ' ;
X         } ;
X/* make everything changed on first update of tmp
                         */
X    touchwin(tmp) ;
X    ERR = 0 ;
X    OK = 1 ;
X    return(tmp) ;
X}
X/* #]newwin:
                              */
X/* #[delwin: delete a window
                              */
Xdelwin(w)
XWINDOW *w ;
X{
X    if ( w == NULL )
X         {
X         ERR = 1 ;
X         OK = 0 ;
X         return ;
X         } ;
X    if ( w->_flags & _SUBWIN )
X         free(w->_y) ;
X    else
X         {
X         free(w->_y[0]) ;
X         free(w->_y) ;
X         } ;
X    free(w) ;
X    ERR = 0 ;
X    OK = 1 ;
X}
X/* #]delwin:
                              */
X/* #[mvwin: move window
                                   */
Xmvwin(w,y,x)
XWINDOW *w ;
Xint y,x ;
X{
X    if ( y < 0 || x < 0 || ( y + w->_maxy + 1 ) > LINES || ( x + w->_maxx + 1 ) >
COLS )
X         {
X         ERR = 1 ;
X         OK = 0 ;
X         return ;
X         } ;
X    w->_begy = y ;
X    w->_begx = x ;
```

```
X    touchwin(w) ;
X    ERR = 0 ;
X    OK = 1 ;
X}
X/* #]mvwin:
                                    */
X/* #[touchwin: touch a window
                                */
Xtouchwin(w)
XWINDOW *w ;
X{
X    WORD i,j ;
X    w->_firstch = w->_y[0] ;
X    w->_lastch = &(w->_y[w->_maxy][w->_maxx]) ;
X    for ( i = 0 ; i <= w->_maxy ; i++ )
X         for ( j = 0 ; j <= w->_maxx ; j++ )
X              w->_y[i][j] = w->_y[i][j] | TOUCHED ;
X    ERR = 0 ;
X    OK = 1 ;
X}
X/* #]touchwin:
                                    */
X/* #[subwin: create a sub-window
                                */
XWINDOW *subwin(w,l,c,by,bx)
XWINDOW *w ;
Xint l,c,by,bx ;
X{
X    WINDOW *tmp ;
X    WORD   i, nl, nc ;
X    char *malloc() ;
X
X/* cannot take subwindows of curscr
                                */
X    if ( w == curscr )
X         {
X         ERR = 1 ;
X         OK = 0 ;
X         return(NULL) ;
X         } ;
X    tmp = ( WINDOW * ) malloc(sizeof(WINDOW)) ;
X    if ( tmp == NULL )
X         {
X         ERR = 1 ;
X         OK = 0 ;
X         return(NULL) ;
X         } ;
X    tmp->_curx = 0 ;
X    tmp->_cury = 0 ;
X    tmp->_clear = 0 ;
X    tmp->_leave = 0 ;
X    tmp->_scroll = 0 ;
X    if ( l == 0 )
```

```
X        nl = LINES - by ;
X    else
X        nl = l ;
X    if ( c == 0 )
X        nc = COLS - bx ;
X    else
X        nc = c ;
X    if ( l < 1 || l > (w->_maxy+1) || c < 1 || c > (w->_maxx+1) )
X        {
X        free(tmp) ;
X        ERR = 1 ;
X        OK = 0 ;
X        return(NULL) ;
X        } ;
X    tmp->_maxy = nl - 1 ;
X    tmp->_maxx = nc - 1 ;
X    if ( nl == LINES && nc == COLS )
X        tmp->_flags = _FULLWIN | _SUBWIN ;
X    else
X        tmp->_flags = _SUBWIN ;
X    if ( by < w->_begy || by >= (w->_maxy+w->_begy) ||
X         bx < w->_begx || bx >= (w->_maxx+w->_begx) )
X        {
X        free(tmp) ;
X        ERR = 1 ;
X        OK = 0 ;
X        return(NULL) ;
X        } ;
X    tmp->_begy = by ;
X    tmp->_begx = bx ;
X    tmp->_y = ( WORD ** ) malloc(sizeof(WORD *)*(tmp->_maxy+1)) ;
X    if ( tmp->_y == NULL )
X        {
X        free(tmp) ;
X        ERR = 1 ;
X        OK = 0 ;
X        return(NULL) ;
X        } ;
X    tmp->_y[0] = w->_y[0] + ( tmp->_begy - w->_begy ) * ( w->_maxx + 2 ) + ( tmp-
>_maxx - w->_maxx ) ;
X    for ( i = 1 ; i <= tmp->_maxy ; i++ )
X        tmp->_y[i] = tmp->_y[0] + i * ( w->_maxx + 2 ) ;
X/* make everything changed on first update of tmp
                        */
X    touchwin(tmp) ;
X    ERR = 0 ;
X    OK = 1 ;
X    return(tmp) ;
X}
X/* #]subwin:
                              */
X/* #[leaveok: tell curses where to leave the cursor after updating
              */
```

```
Xleaveok(w,f)
XWINDOW *w ;
Xbool f ;
X{
X    w->_leave = f ;
X    ERR = 0 ;
X    OK = 1 ;
X}
X/* #]leaveok:
                                    */
X/* #[scrollok: tell curses it is ok to scroll the window
                           */
Xscrollok(w,f)
XWINDOW *w ;
Xbool f ;
X{
X    w->_scroll = f ;
X    ERR = 0 ;
X    OK = 1 ;
X}
X/* #]scrollok:
                                    */
X/* #[nl: tell curses to map CR to CR,LF
                                */
Xnl()
X{
X    _nlmap = 1 ;
X    ERR = 0 ;
X    OK = 1 ;
X}
X/* #]nl:
                                */
X/* #[nonl: tell curses not to map CR to CR,LF
                           */
Xnonl()
X{
X    _nlmap = 0 ;
X    ERR = 0 ;
X    OK = 1 ;
X}
X/* #]nonl:
                                    */
X/* #[longname: return the full name of the terminal                          */
X/* always returns the contents of ttytype in name
                           */
Xlongname(termbuf,name)
Xchar *termbuf, *name ;
X{
X    int i ;
X
X    for ( i = 0 ; ttytype[i] != '\0' ; i++ )
X         name[i] = ttytype[i] ;
X    name[i] = '\0' ;
```

```
X}
X/* #]longname:
                                  */
X/* #]miscel:
                               */
X/* #[output: output functions
                               */
X/* #[waddch: add a character to a window
                         */
Xwaddch(w,c)
XWINDOW *w ;
Xchar   c ;
X{
X    WORD i, tpos ;
X
X    ERR = 0 ;
X    OK = 1 ;
X    if ( c >= 0x20 )
X         {
X         if ( w == curscr )
X              {
X              if ( ( w->_flags & _STANDOUT ))
X                   {
X                   w->_y[w->_cury][w->_curx] = c | STANDOUT | TOUCHED ;
X                   Bconout(2,'\033') ;
X                   Bconout(2,'p') ;
X                   Bconout(5,c) ;
X                   Bconout(2,'\033') ;
X                   Bconout(2,'q') ;
X                   }
X              else
X                   {
X                   w->_y[w->_cury][w->_curx] = c | TOUCHED ;
X                   Bconout(5,c) ;
X                   } ;
X              _csrx++ ;
X              }
X         else
X              {
X              if ( (w->_flags & _STANDOUT ))
X                   w->_y[w->_cury][w->_curx] = c | STANDOUT | TOUCHED ;
X              else
X                   w->_y[w->_cury][w->_curx] = c | TOUCHED ;
X              if ( w->_firstch == 0 )
X                   {
X                   w->_firstch = &(w->_y[w->_cury][w->_curx]) ;
X                   w->_lastch = w->_firstch ;
X                   }
X              else
X                   {
X                   if ( w->_firstch > &(w->_y[w->_cury][w->_curx]) )
X                        w->_firstch = &(w->_y[w->_cury][w->_curx]) ;
X                   else
```

```
X                    {
X                    if ( w->_lastch < &(w->_y[w->_cury][w->_curx]) )
X                         w->_lastch = &(w->_y[w->_cury][w->_curx]) ;
X                    } ;
X               } ;
X          } ;
X        wmove(w,w->_cury,w->_curx+1) ;
X        }
X    else if ( c == '\n' )
X        {
X        wclrtoeol(w) ;
X        if ( _nlmap )
X             wmove(w,w->_cury+1,0) ;
X        else
X             wmove(w,w->_cury+1,w->_curx) ;
X        }
X    else if ( c == '\r' )
X        {
X        wmove(w,w->_cury,0) ;
X        }
X    else if ( c == '\t' )
X        {
X        tpos = ( w->_curx + w->_begx ) % 8 ;
X        tpos = ( tpos == 0 ) ? 8 : tpos ;
X        for ( i = 0 ; i < tpos ; i++ )
X             waddch(w,' ') ;
X        }
X    else
X        {
X        if ( w == curscr )
X             {
X             if ( ( w->_flags & _STANDOUT ))
X                  {
X                  w->_y[w->_cury][w->_curx] = c | STANDOUT | TOUCHED ;
X                  Bconout(2,'\033') ;
X                  Bconout(2,'p') ;
X                  Bconout(5,c) ;
X                  Bconout(2,'\033') ;
X                  Bconout(2,'q') ;
X                  }
X             else
X                  {
X                  w->_y[w->_cury][w->_curx] = c | TOUCHED ;
X                  Bconout(5,c) ;
X                  } ;
X             _csrx++ ;
X             }
X        else
X             {
X             if ( (w->_flags & _STANDOUT ))
X                  w->_y[w->_cury][w->_curx] = c | STANDOUT | TOUCHED ;
X             else
X                  w->_y[w->_cury][w->_curx] = c | TOUCHED ;
```

```
X             if ( w->_firstch == 0 )
X                  {
X                  w->_firstch = &(w->_y[w->_cury][w->_curx]) ;
X                  w->_lastch = w->_firstch ;
X                  }
X             else
X                  {
X                  if ( w->_firstch > &(w->_y[w->_cury][w->_curx]) )
X                       w->_firstch = &(w->_y[w->_cury][w->_curx]) ;
X                  else
X                       {
X                       if ( w->_lastch < &(w->_y[w->_cury][w->_curx]) )
X                            w->_lastch = &(w->_y[w->_cury][w->_curx]) ;
X                       } ;
X                  } ;
X             } ;
X         wmove(w,w->_cury,w->_curx+1) ;
X         } ;
X}
X/* #]waddch:
                             */
X/* #[waddstr: add a string of characters to a window
                        */
Xwaddstr(w,s)
XWINDOW *w ;
Xchar   *s ;
X{
X    WORD  i ;
X
X    ERR = 0 ;
X    OK = 1 ;
X    for ( i = 0 ; s[i] != '\0' ; i++ )
X         waddch(w,s[i]) ;
X}
X/* #]waddstr:
                                  */
X/* #[box: draw a box around a window
                        */
Xbox(w,v,h)
XWINDOW *w ;
Xchar v,h ;
X{
X    WORD i ;
X
X    ERR = 0 ;
X    OK = 1 ;
X    for ( i = 0 ; i <= w->_maxy ; i++ )
X         {
X         mvwaddch(w,i,0,v) ;
X         mvwaddch(w,i,w->_maxx,v) ;
X         } ;
X    for ( i = 1 ; i < w->_maxx ; i++ )
X         {
```

```
X         mvwaddch(w,0,i,h) ;
X         mvwaddch(w,w->_maxy,i,h) ;
X         } ;
X}
X/* #]box:
                                  */
X/* #[wclear: clear a window
                                  */
Xwclear(w)
XWINDOW *w ;
X{
X    ERR = 0 ;
X    OK = 1 ;
X    werase(w) ;
X    clearok(w,TRUE) ;
X    w->_curx = 0 ;
X    w->_cury = 0 ;
X}
X/* #]wclear:
                              */
X/* #[wclrtobo: clear a window to the bottom
                              */
Xwclrtobot(w)
XWINDOW *w ;
X{
X    WORD i,j ;
X
X    ERR = 0 ;
X    OK = 1 ;
X    for ( i = w->_curx ; i <= w->_maxx ; i++ )
X         w->_y[w->_cury][i] = ' ' | TOUCHED ;
X    for ( i = w->_cury + 1 ; i <= w->_maxy ; i++ )
X         {
X         for ( j = 0 ; j <= w->_maxx ; j++ )
X              w->_y[i][j] = ' ' | TOUCHED ;
X         } ;
X    if ( w == curscr )
X         {
X         Bconout(2,'\033') ;
X         Bconout(2,'J') ;
X         }
X    else
X         {
X         if ( w->_firstch == 0 )
X              {
X              w->_firstch = &(w->_y[w->_cury][w->_curx]) ;
X              w->_lastch = &(w->_y[w->_maxy][w->_maxx]) ;
X              }
X         else
X              {
X              if ( w->_firstch > &(w->_y[w->_cury][w->_curx]) )
X                   w->_firstch = &(w->_y[w->_cury][w->_curx]) ;
X              w->_lastch = &(w->_y[w->_maxy][w->_maxx]) ;
```

```
X             } ;
X        } ;
X}
X/* #]wclrtobo:
                                    */
X/* #[wclrtoeo: clear a window to the end of the line
                         */
Xwclrtoeol(w)
XWINDOW *w ;
X{
X    WORD i ;
X
X    ERR = 0 ;
X    OK = 1 ;
X    for ( i = w->_curx ; i <= w->_maxx ; i++ )
X         w->_y[w->_cury][i] = ' ' | TOUCHED ;
X    if ( w == curscr )
X         {
X         Bconout(2,'\033') ;
X         Bconout(2,'K') ;
X         }
X    else
X         {
X         if ( w->_firstch == 0 )
X              {
X              w->_firstch = &(w->_y[w->_cury][w->_curx]) ;
X              w->_lastch = &(w->_y[w->_cury][w->_maxx]) ;
X              }
X         else
X              {
X              if ( w->_firstch > &(w->_y[w->_cury][w->_curx]) )
X                   w->_firstch = &(w->_y[w->_cury][w->_curx]) ;
X              if ( w->_lastch < &(w->_y[w->_cury][w->_maxx]) )
X                   w->_lastch = &(w->_y[w->_cury][w->_maxx]) ;
X              } ;
X         } ;
X}
X/* #]wclrtoeo:
                                    */
X/* #[wdelch: delete a character on a window
                               */
Xwdelch(w)
XWINDOW *w ;
X{
X    WORD ox, oy, i ;
X
X    ERR = 0 ;
X    OK = 1 ;
X    ox = w->_curx ;
X    oy = w->_cury ;
X    for ( i = ox + 1 ; i <= w->_maxx ; i++ )
X         {
X         w->_y[oy][i-1] = w->_y[oy][i] | TOUCHED ;
```

```
X        } ;
X    w->_y[oy][w->_maxx] = ' ' | TOUCHED ;
X    w->_curx = ox ;
X    w->_cury = oy ;
X    if ( w == curscr )
X        {
X        Bconout(2,'\033') ;
X        Bconout(2,'K') ;
X        for ( i = w->_curx ; i <= w->_maxx ; i++ )
X            {
X            Bconout(5,(w->_y[w->_cury][i] & 0xff )) ;
X            _csrx++ ;
X            } ;
X        _movcur(w->_cury,w->_curx) ;
X        }
X    else
X        {
X        if ( w->_firstch == 0 )
X            {
X            w->_firstch = &(w->_y[w->_cury][w->_curx]) ;
X            w->_lastch = &(w->_y[w->_cury][w->_maxx]) ;
X            }
X        else
X            {
X            if ( w->_firstch > &(w->_y[w->_cury][w->_curx]) )
X                w->_firstch = w->_y[w->_cury] ;
X            if ( w->_lastch < &(w->_y[w->_cury][w->_maxx]) )
X                w->_lastch = &(w->_y[w->_cury][w->_maxx]) ;
X            } ;
X        } ;
X}
X/* #]wdelch:
                            */
X/* #[wdeletel: delete a line from a window
                            */
Xwdeleteln(w)
XWINDOW *w ;
X{
X    WORD i,j ;
X
X    ERR = 0 ;
X    OK = 1 ;
X    for ( i = w->_cury + 1 ; i <= w->_maxy ; i++ )
X        {
X        for ( j = 0 ; j <= w->_maxx ; j++ )
X            w->_y[i-1][j] = w->_y[i][j] | TOUCHED ;
X        } ;
X    for ( j = 0 ; j <= w->_maxx ; j++ )
X        w->_y[w->_maxy][j] = ' ' | TOUCHED ;
X    if ( w == curscr )
X        {
X        Bconout(2,'\033') ;
X        Bconout(2,'M') ;
```

```
X         _csrx = 0 ;
X         _movcur(w->_cury,w->_curx) ;
X         }
X    else
X         {
X         if ( w->_firstch == 0 )
X              {
X              w->_firstch = w->_y[w->_cury] ;
X              w->_lastch = &(w->_y[w->_maxy][w->_maxx]) ;
X              }
X         else
X              {
X              if ( w->_firstch > w->_y[w->_cury] )
X                   w->_firstch = w->_y[w->_cury] ;
X              w->_lastch = &(w->_y[w->_maxy][w->_maxx]) ;
X              } ;
X         } ;
X}
X/* #]wdeletel:
                                   */
X/* #[werase: erase a window
                                   */
Xwerase(w)
XWINDOW *w ;
X{
X    WORD i,j ;
X
X    ERR = 0 ;
X    OK = 1 ;
X    for ( i = 0 ; i <= w->_maxy ; i++ )
X         {
X         for ( j = 0 ; j <= w->_maxx ; j++ )
X              w->_y[i][j] = ' ' | TOUCHED ;
X         } ;
X    if ( w == curscr )
X         {
X         Bconout(2,'\033') ;
X         Bconout(2,'E') ;
X         _csry = 0 ;
X         _csrx = 0 ;
X         _movcur(curscr->_cury,curscr->_curx) ;
X         }
X    else
X         {
X         w->_firstch = w->_y[0] ;
X         w->_lastch = &(w->_y[w->_maxy][w->_maxx]) ;
X         } ;
X}
X/* #]werase:
                               */
X/* #[winsch: insert a character
                               */
Xwinsch(w,c)
```

```
XWINDOW *w ;
Xchar c ;
X{
X    WORD ox,oy,i ;
X    BYTE tstr[2] ;
X
X    ERR = 0 ;
X    OK = 1 ;
X    ox = w->_curx ;
X    oy = w->_cury ;
X    for ( i = w->_maxx - 1 ; i >= ox ; --i )
X         {
X         w->_y[oy][i+1] = w->_y[oy][i] | TOUCHED ;
X         } ;
X    if ( w == curscr )
X         {
X         Bconout(2,'\033') ;
X         Bconout(2,'K') ;
X         for ( i = w->_curx ; i <= w->_maxx ; i++ )
X              {
X              Bconout(5,(w->_y[w->_cury][i] & 0xff )) ;
X              _csrx++ ;
X              } ;
X         _movcur(w->_cury,w->_curx) ;
X         }
X    else
X         {
X         if ( w->_firstch == 0 )
X              {
X              w->_firstch = &(w->_y[w->_cury][w->_curx]) ;
X              w->_lastch = &(w->_y[w->_cury][w->_maxx]) ;
X              }
X         else
X              {
X              if ( w->_firstch > &(w->_y[w->_cury][w->_curx]) )
X                   w->_firstch = &(w->_y[w->_cury][w->_curx]) ;
X              if ( w->_lastch < &(w->_y[w->_cury][w->_maxx]) )
X                   w->_lastch = &(w->_y[w->_cury][w->_maxx]) ;
X              } ;
X         } ;
X    mvwaddch(w,oy,ox,c) ;
X}
X/* #]winsch:
                              */
X/* #[winsertl: insert a line in a window
                         */
Xwinsertln(w)
XWINDOW *w ;
X{
X    WORD ox,oy,i,j ;
X
X    ERR = 0 ;
X    OK = 1 ;
```

```
X    for ( i = w->_maxy - 1 ; i >= w->_cury ; --i )
X         {
X         for ( j = 0 ; j <= w->_maxx ; j++ )
X              w->_y[i+1][j] = w->_y[i][j] | TOUCHED ;
X         } ;
X    for ( j = 0 ; j <= w->_maxx ; j++ )
X         w->_y[w->_cury][j] = ' ' | TOUCHED ;
X    if ( w == curscr )
X         {
X         Bconout(2,'\033') ;
X         Bconout(2,'L') ;
X         _csrx = 0 ;
X         _movcur(w->_cury,w->_curx) ;
X         }
X    else
X         {
X         if ( w->_firstch == 0 )
X              w->_firstch = w->_y[w->_cury] ;
X         else
X              {
X              if ( w->_firstch > w->_y[w->_cury] )
X                   w->_firstch = w->_y[w->_cury] ;
X              } ;
X         w->_lastch = &(w->_y[w->_maxy][w->_maxx]) ;
X         } ;
X}
X/* #]winsertl:
                                   */
X/* #[wmove: move the cursor of the window to a location
                              */
Xwmove(w,y,x)
XWINDOW *w ;
Xint y,x ;
X{
X    WORD i ;
X
X    ERR = 0 ;
X    OK = 1 ;
X    if ( x < 0 )
X         {
X         w->_curx = 0 ;
X         ERR = 1 ;
X         OK = 0 ;
X         }
X    else
X         {
X         if ( x > w->_maxx )
X              {
X              w->_curx = w->_maxx ;
X              ERR = 1 ;
X              OK = 0 ;
X              }
X         else
```

```
X             w->_curx = x ;
X        } ;
X    if ( y < 0 )
X        {
X        w->_cury = 0 ;
X        ERR = 1 ;
X        OK = 0 ;
X        }
X    else
X        {
X        if ( y > w->_maxy )
X             {
X             if ( w->_scroll )
X                  {
X                  for ( i = w->_maxy ; i < y ; i++ )
X                       scroll(w) ;
X                  } ;
X             w->_cury = w->_maxy ;
X             ERR = 1 ;
X             OK = 0 ;
X             }
X        else
X             w->_cury = y ;
X        } ;
X    if ( w == curscr )
X        {
X        _movcur(curscr->_cury,curscr->_curx) ;
X        } ;
X}
X/* #]wmove:
                                  */
X/* #[woverlay: overlay two windows.
                               */
Xwoverlay(v,w)
XWINDOW *v, *w ;
X{
X    WORD i, j ;
X
X    for ( i = 0 ; i <= w->_maxy && i <= v->_maxy ; i++ )
X        {
X        for ( j = 0 ; j <= w->_maxx && j <= v->_maxx ; j++ )
X             {
X             if ( ( v->_y[i][j] & 0xff ) == 0x20 )
X                  w->_y[i][j] = v->_y[i][j] | TOUCHED ;
X             } ;
X        } ;
X    if ( w == curscr )
X        wrefresh(curscr) ;
X    else
X        {
X        w->_firstch = w->_y[0] ;
X        w->_lastch = &(w->_y[w->_maxy][w->_maxx]) ;
X        } ;
```

```
X}
X/* #]woverlay:
                                */
X/* #[overwrit: overwrite two windows.
                               */
Xoverwrite(v,w)
XWINDOW *v, *w ;
X{
X    WORD i, j ;
X
X    for ( i = 0 ; i <= w->_maxy && i <= v->_maxy ; i++ )
X         {
X         for ( j = 0 ; j <= w->_maxx && j <= v->_maxx ; j++ )
X              {
X              w->_y[i][j] = v->_y[i][j] | TOUCHED ;
X              } ;
X         } ;
X    if ( w == curscr )
X         wrefresh(curscr) ;
X    else
X         {
X         w->_firstch = w->_y[0] ;
X         w->_lastch = &(w->_y[w->_maxy][w->_maxx]) ;
X         } ;
X}
X/* #]overwrit:
                                */
X/* #[wstandou: set the standout flag for a window
                           */
Xwstandout(w)
XWINDOW *w ;
X{
X    w->_flags = w->_flags | _STANDOUT ;
X    ERR = 0 ;
X    OK = 1 ;
X}
X/* #]wstandou:
                                */
X/* #[wstanden: end standout mode
                               */
Xwstandend(w)
XWINDOW *w ;
X{
X    w->_flags = w->_flags & ( ~_STANDOUT) ;
X    ERR = 0 ;
X    OK = 1 ;
X}
X/* #]wstanden:
                                */
X/* #]output:
                               */
X/* #[input:  input functions
                                */
```

```
X/* #[raw: set terminal in raw mode
                              */
Xraw()
X{
X/*
X * raw mode means :
X * when getting a character from the keyboard, return everything
X * including keyboard shift state.
X */
X  int    __sshft() ;
X
X  if ( _modinp == RAW )
X    return ;
X  Supexec(__sshft) ;
X  _modinp = RAW ;
X}
X/* #]raw:
                                    */
X/* #[noraw: reset terminal from raw mode into cooked mode
                         */
Xnoraw()
X{
X    int __ushft() ;
X    if ( _modinp != RAW )
X         return ;
X    Supexec(__ushft) ;
X    _modinp = COOKED ;
X}
X/* #]noraw:
                                    */
X/* #[__sshft: set terminal to return keyboard shift state
                         */
X/* execute in supervisor mode only
                              */
X__sshft()
X{
X    int ov, nv ;
X
X    ov = CONTERM ;
X    nv = ov | 0x08 ;
X    CONTERM = nv ;
X}
X/* #]__sshft:
                                    */
X/* #[__ushft: set terminal to not return keyboard shift state
                   */
X/* execute in supervisor mode only
                              */
X__ushft()
X{
X    int ov, nv ;
X
X    ov = CONTERM ;
```

```
X    nv = ov & ~0x08 ;
X    CONTERM = nv ;
X}
X/* #]__ushft:
                                    */
X/* #[crmode: set terminal in cbreak mode
                         */
X/*
X * cbreak mode means for the atari ;
X * return both keyboard scan code as well as the character value.
X * kill the process with a process terminate -1 as soons as a control C is met
X * control D is also recognized to be the EOF.
X */
Xcrmode()
X{
X    if ( _modinp == CBREAK )
X         return ;
X    if ( _modinp == RAW )
X         noraw() ;
X    _modinp = CBREAK ;
X}
X/* #]crmode:
                               */
X/* #[nocrmode: reset terminal from cbreak into cooked mode
                         */
Xnocrmode()
X{
X    if ( _modinp != CBREAK )
X         return ;
X    _modinp = COOKED ;
X}
X/* #]nocrmode:
                                    */
X/* #[echo: set curses to echo characters on input
                         */
Xecho()
X{
X    _doecho = TRUE ;
X    ERR = 0 ;
X    OK = 1 ;
X}
X/* #]echo:
                                    */
X/* #[noecho: set curses not to echo characters on input
                               */
Xnoecho()
X{
X    _doecho = FALSE ;
X    ERR = 0 ;
X    OK = 1 ;
X}
X/* #]noecho:
                               */
```

```
X/* #[wgetch: get a character from the terminal
                            */
X/* WARNING : wgetch returns a 32-bit value, not a char although only
         */
X/* the lowest 8 bits may actually be transmitted.
                         */
Xwgetch(w)
XWINDOW *w ;
X{
X    bool reset ;
X    LONG retval ;
X
X    reset = FALSE ;
X    if ( _modinp == COOKED && _doecho == TRUE )
X         {
X         reset = TRUE ;
X         crmode() ;
X         } ;
X    Bconout(2,'\033') ;
X    Bconout(2,'e') ;
X    switch ( _modinp )
X         {
X         case COOKED:
X              while ( !Cconis() ) ;
X              retval = Cnecin() ;
X              retval = retval & 0x00ff ;
X/* kill process on control C
                            */
X              if ( retval == 0x03 )
X                   Pterm(-1L) ;
X              break ;
X         case CBREAK:
X              while ( !Cconis() ) ;
X              retval = Cnecin() ;
X/* kill process on control C
                            */
X              if ( ( retval & 0x00ff ) == 0x03 )
X                   Pterm(-1L) ;
X              break ;
X         case RAW:
X              while ( !Cconis() ) ;
X              retval = Crawcin() ;
X              break ;
X         } ;
X    Bconout(2,'\033') ;
X    Bconout(2,'f') ;
X    if ( _doecho )
X         waddch(w,(char ) ( 0x00ff & retval )) ;
X    if ( reset )
X         nocrmode() ;
X    return(retval) ;
X}
X/* #]wgetch:
```

```
                              */
X/* #[wgetstr: get a string from the terminal
                         */
Xwgetstr(w,s)
XWINDOW *w ;
Xchar *s ;
X{
X    WORD ox, oy ;
X    bool reset, end ;
X    char c ;
X    int i ;
X    LONG wgetch() ;
X
X    reset = FALSE ;
X    getyx(w,oy,ox) ;
X    if ( _modinp == COOKED && _doecho == TRUE )
X         {
X         reset = TRUE ;
X         _doecho = FALSE ;
X         } ;
X    i = 0 ;
X    for ( end = FALSE ; !end ; i++ )
X         {
X         wrefresh(w);
X         switch ( _modinp )
X              {
X              case COOKED:
X                   c = ( char ) wgetch(w) ;
X                   if ( c == 0x0d || c == 0x0a || c == 0x04 || c == 0 )
X                        {
X                        s[i] = '\0' ;
X                        end = TRUE ;
X                        break ;
X                        } ;
X/* receive a backspace */
X                   if ( c == 0x08 ) {
X                        if ( i != 0 )
X                             {
X                             --i ;
X                             s[i] = 0 ;
X                             if ( reset ) {
X                                  int  x, y;
X                                  getyx(w, y, x);
X                                  mvwaddch(w,y,x-1,' ');
X                                  wmove(w, y, x-1);
X/*
X                                  mvwaddstr(w,oy,ox,s) ;
X*/
X                             }
X                        }
X                        i--;
X                        break;
X                   }
```

```
X/* receive control U or line kill
                              */
X                  if ( c == 0x13 )
X                       {
X                       i = 0 ;
X                       if ( reset )
X                            wmove(w,oy,ox) ;
X                       break ;
X                       } ;
X                  s[i] = c ;
X                  if (reset)
X                       waddch(w,c) ;
X                  break ;
X             case CBREAK:
X                  c = ( char ) wgetch(w) ;
X                  if ( c == 0x0d || c == 0x0a || c == 0x04 || c == 0 )
X                       {
X                       s[i] = '\0' ;
X                       end = TRUE ;
X                       break ;
X                       } ;
X                  s[i] = c ;
X                  break ;
X             case RAW:
X                  c = ( char ) wgetch(w) ;
X                  if ( c == 0x0d || c == 0x0a || c == 0x04 || c == 0 )
X                       {
X                       s[i] = '\0' ;
X                       end = TRUE ;
X                       break ;
X                       } ;
X                  s[i] = c ;
X                  break ;
X             } ;
X         } ;
X    if ( reset )
X         _doecho = TRUE ;
X}
X/* #]wgetstr:
                                   */
X/* #]input:
                              */
X/* #[wrefresh: refresh a window on the screen
                         */
Xwrefresh(w)
XWINDOW *w ;
X{
X    WORD i,j,k,l ;
X    WORD c, *ptr ;
X
X    ERR = 0 ;
X    OK = 1 ;
X    if ( w != curscr && curscr->_clear )
```

```
X        {
X        Bconout(2,'\033') ;
X        Bconout(2,'E') ;
X        _csry = 0 ;
X        _csrx = 0 ;
X        for ( i = 0 ; i < LINES ; i++ )
X             {
X             for ( j = 0 ; j < COLS ; j++ )
X                  curscr->_y[i][j] = ' ' ;
X             } ;
X        curscr->_clear = 0 ;
X        } ;
X    if ( w->_clear )
X        {
X        if ( ( w->_flags & _FULLWIN ) )
X             {
X             Bconout(2,'\033') ;
X             Bconout(2,'E') ;
X             _csry = 0 ;
X             _csrx = 0 ;
X             for ( i = 0 ; i < LINES ; i++ )
X                  {
X                  for ( j = 0 ; j < COLS ; j++ )
X                       curscr->_y[i][j] = ' ' ;
X                  } ;
X             } ;
X        w->_firstch = w->_y[0] ;
X        w->_lastch = &(w->_y[w->_maxy][w->_maxx]) ;
X        w->_clear = 0 ;
X        } ;
X    if ( w != curscr )
X        {
X        if ( w->_firstch != 0 )
X             {
X             if ( w->_flags & _SUBWIN )
X                  {
X                  for ( i = 0 ; i <= w->_maxy ; i++ )
X                       {
X                       ptr = w->_y[i] ;
X                       if ( ptr >= w->_firstch && ptr <= w->_lastch )
X                            {
X                            for ( j = 0 ; j <= w->_maxx ; j++ )
X                                 {
X                                 c = ptr[j] ;
X                                 k = i + w->_begy ;
X                                 l = j + w->_begx ;
X                                 if ( ( c & TOUCHED ) && ( k >= 0 && k < LINES &&
l >= 0 && l < COLS ) )
X                                      {
X                                      ptr[j] = c & ~TOUCHED ;
X                                      if ( ( curscr->_y[k][l] & 0x01ff ) != ( c &
0x01ff ) )
X                                           {
```

```
X                                        curscr->_y[k][l] = c ;
X                                        _movcur(k,l) ;
X                                        if ( ( c & STANDOUT ) )
X                                             {
X                                             Bconout(2,'\033') ;
X                                             Bconout(2,'p') ;
X                                             Bconout(5,(c & 0xff)) ;
X                                             Bconout(2,'\033') ;
X                                             Bconout(2,'q') ;
X                                             }
X                                        else
X                                             {
X                                             Bconout(5,( c & 0xff )) ;
X                                             } ;
X                                        _csry = k ;
X                                        _csrx = l + 1 ;
X                                        } ;
X                                   } ;
X                              } ;
X                         } ;
X                    }
X               else
X                    {
X                    for ( ptr = w->_firstch ; ptr <= w->_lastch ; ptr++ )
X                         {
X                         c = *ptr ;
X                         if ( c & TOUCHED )
X                              {
X                              k = ( WORD ) ( ptr - w->_y[0] ) ;
X                              k = k / (  w->_maxx + 1 ) ;
X                              l = ( WORD ) ( ptr - w->_y[k] ) + w->_begx ;
X                              k = k + w->_begy ;
X                              if ( k >= 0 && k < LINES && l >= 0 && l < COLS )
X                                   {
X                                   *ptr = c & ~TOUCHED ;
X                                   if ( ( curscr->_y[k][l] & 0x01ff ) != ( c &
0x01ff ) )
X                                        {
X                                        curscr->_y[k][l] = c ;
X                                        _movcur(k,l) ;
X                                        if ( ( c & STANDOUT ) )
X                                             {
X                                             Bconout(2,'\033') ;
X                                             Bconout(2,'p') ;
X                                             Bconout(5,(c & 0xff)) ;
X                                             Bconout(2,'\033') ;
X                                             Bconout(2,'q') ;
X                                             }
X                                        else
X                                             {
X                                             Bconout(5,( c & 0xff )) ;
X                                             } ;
```

```
X                                   _csry = k ;
X                                   _csrx = l + 1 ;
X                                   } ;
X                              } ;
X                         } ;
X                    } ;
X               } ;
X               w->_firstch = 0 ;
X               w->_lastch = 0 ;
X               if ( w->_leave )
X                    {
X                    w->_curx = _csrx - w->_begx ;
X                    w->_cury = _csry - w->_begy ;
X                    curscr->_cury = _csry ;
X                    curscr->_curx = _csrx ;
X                    }
X               else
X                    {
X                    curscr->_cury = w->_cury + w->_begy ;
X                    curscr->_curx = w->_curx + w->_begx ;
X                    _movcur(curscr->_cury, curscr->_curx) ;
X                    } ;
X               }
X          else
X               {
X               curscr->_cury = w->_cury + w->_begy ;
X               curscr->_curx = w->_curx + w->_begx ;
X               _movcur(curscr->_cury, curscr->_curx) ;
X               } ;
X          }
X     else
X          {
X          Bconout(2,'\033') ;
X          Bconout(2,'H') ;
X          _csry = 0 ;
X          _csrx = 0 ;
X          for ( i = 0 ; i < LINES ; i++ )
X               {
X               for ( j = 0 ; j < COLS ; j++ )
X                    {
X                    c = w->_y[i][j] ;
X                    if ( ( c & STANDOUT ) )
X                         {
X                         Bconout(2,'\033') ;
X                         Bconout(2,'p') ;
X                         Bconout(5,(c & 0xff)) ;
X                         Bconout(2,'\033') ;
X                         Bconout(2,'q') ;
X                         }
X                    else
X                         {
X                         Bconout(5,(c & 0xff)) ;
X                         } ;
```

```
X                   _csrx++ ;
X                   } ;
X               _movcur(i+1,0) ;
X               } ;
X           _movcur( curscr->_cury, curscr->_curx) ;
X           } ;
X}
X/* #]wrefresh:
                                    */
X/* #[detail: detail functions of curses
                                */
X/* #[mvcur: move cursor in standard curses manner
                            */
Xmvcur(ly,lx,ny,nx)
Xint ly,lx,ny,nx ;
X{
X    _movcur((WORD) ny,( WORD) nx) ;
X}
X/* #]mvcur:
                                    */
X/* #[_movcur: move cursor
                                */
X_movcur(y,x)
XWORD y,x ;
X{
X    if ( _csry == y && _csrx == x )
X        return ;
X    Bconout(2,'\033') ;
X    Bconout(2,'Y') ;
X    Bconout(2,y+' ') ;
X    Bconout(2,x+' ') ;
X    _csry = y ;
X    _csrx = x ;
X}
X/* #]_movcur:
                                    */
X/* #[scroll: scroll a window upward one line
                            */
Xscroll(w)
XWINDOW *w ;
X{
X    WORD i,j ;
X
X    ERR = 0 ;
X    OK = 1 ;
X    for ( i = 0 ; i < w->_maxy ; i++ )
X        for ( j = 0 ; j <= w->_maxx ; j++ )
X            w->_y[i][j] = w->_y[i+1][j] ;
X    for ( i = 0 ; i <= w->_maxx ; i++ )
X        w->_y[w->_maxy][i] = ' ' ;
X    if ( w == curscr )
X        wrefresh(curscr) ;
X    else
```

```
X        {
X        w->_firstch = w->_y[0] ;
X        w->_lastch = &(w->_y[w->_maxy][w->_maxx]) ;
X        } ;
X}
X/* #]scroll:
                              */
X/* #]detail:
                              */
X/* #]curses:
                              */
X
X/*
X * Extra stuff added - tony
X */
X
Xprintw(format, a1, a2, a3, a4, a5, a6, a7, a8, a9)
Xchar    *format;
Xint a1, a2, a3, a4, a5, a6, a7, a8, a9;
X{
X    char lbuf[256];
X
X    sprintf(lbuf, format, a1, a2, a3, a4, a5, a6, a7, a8, a9);
X    addstr(lbuf);
X}
X
Xscanw(format, a1, a2, a3, a4, a5, a6, a7, a8, a9)
Xchar    *format;
Xint a1, a2, a3, a4, a5, a6, a7, a8, a9;
X{
X    char lbuf[256];
X
X    echo();
X    getstr(lbuf);
X    sscanf(lbuf, format, a1, a2, a3, a4, a5, a6, a7, a8, a9);
X}
SHAR_EOF
chmod 0640 curses.c || echo "restore of curses.c fails"
sed 's/^X//' << 'SHAR_EOF' > curses.doc &&
X           The CURSES screen updating and cursor movement package.
X
X          A collection of C-functions for Atari ST-series computers
X
X                                Version 1.0
X
X
X                              R. van 't Veen
X
X
X
X
X    This document describes a set of C-functions which allow the user
Xeasier design of full-screen applications. This package is intended for
```

Xprograms that perform graphics-like operations on terminal screens.
XIt is largely compatible with the UN*X curses(3x) package. This document
Xprovides pointers to its use, a little detail about its inner workings and it
Xstresses differences between the UN*X-implementation and the Atari-ST series
Ximplementation.
X
X
X
X
X
X
X   Introduction.
X
X   This package was born out of my desire to make more programs available for
XUN*X also available on my Atari. A side-effect of this is that it provides
Xeasier screen updating than by means of sending escape codes. The package is
Xalso quite fast, timing of a sample application yielded a rate of about
X90 kilobaud per second.
X
X   Terminology.
X
X   Throughout this document the following words will be used frequently :
X
XWINDOW   An internal representation of what a section of the terminal
X   screen may eventually look like. A window is a rectangular
X   array, which may be as big as the terminal screen or even as
X   small as a single character.
X
XTERMINAL   Or terminal screen, the packages idea of what the Atari's VT52
X   emulation looks like. If everything is allright this is what you
X   see. The terminal screen is a special screen.
X
XSCREEN   A screen is an element of the set of windows that are as large as
X   the terminal screen. One screen, called stdscr, is already
X   provided for.
X
X   Using the package.
X
X   To use the package, include the file "curses.h" in your source. This gives
Xyou the types and definitions necessary to be able to use it. After compilation
Xone should link with the file containing the compiled curses-source, as well as
Xwith the object-file or library giving you access to the system calls,
Xgemdos, bios and xbios. You also should link in a file giving you access
Xto the malloc() and free() calls. (Note : this is not the GEMDOS Malloc/Mfree
Xpair, but a malloc() and free(), which should be contained in any decent
Xstandard C-function library. GEMDOS Malloc/Mfree are very buggy indeed.)
XThere is a difference with UN*X here : the UN*X-curses includes stdio.h
Xand sgtty.h, whereas Atari-curses includes portab.h and stdio.h. This should
Xgive you the definitions for BYTE, WORD and LONG.
X Some compilers supply default startup files that don't supply enough heap space
Xto run curses in, you should adapt these startup files to your liking.
X
X   Installing the package.

X
X   Compile the file "curses.c" with your favorite C-compiler, to create an
Xobject-file or library, whatever you prefer. Curses.c includes "portab.h"
Xfor BYTE, WORD, LONG, and NULL definitions, osbind.h for system-call macros
Xand curses.h for its own definitions. Remember curses.h also includes stdio.h.
XThe code should be portable between various brands of compilers. Forget about
Xwarnings like : warning : pointer substraction yields a long result.
XNot all compilers adhere to Kernighan and Ritchie standards in this respect.
XThe meaning of each of the constants is given in the table below. In another
Xtable the used system call macro's are given.
X
X+---------------+---+
X| constant name | meaning |
X+===============+===+
X| BYTE | a type with at least 8 bits in it, usually a char |
X| WORD | a type with at least 16 bits in it |
X| LONG | a type with at least 32 bits, used for pointers |
X| NULL | a value assigned to pointers pointing nowhere |
X+---------------+---+
X
X+---------------+---+
X| macro name | meaning |
X+===============+===+
X| Pterm | gemdos $4c, terminate process and return to caller |
X| Cconis | gemdos $0b, console character available ? |
X| Crawcin | gemdos $07, get a raw character from console |
X| Cnecin | gemdos $08, get a not-so raw character from console |
X| Bconout | bios $03, output a character to a device |
X| Bconstat | bios $01, get input device status |
X| Bconin | bios $02, input a character from a device |
X| Supexec | xbios $26, execute function in supervisor mode |
X+---------------+---+
X
X   Screen updating.
X
X   The package defines a data type called WINDOW, which represents what a
Xsection of the terminal may eventually look like. It can be used as a
Xscratch-pad onto which you can make changes. Only after calling
Xthe function refresh() or wrefresh() an attempt will be made by the package
Xto make things on the screen look like the things you specified in the
Xwindow. Information contained in the window include not only its contents,
Xbut also its size, its starting point and various other information. Two
Xstandard window are defined after initialization of the package, namely
Xcurscr, which constitutes the package's idea of what the screen looks like,
Xand stdscr, which is a default window to make changes on. It is advised not
Xto address curscr directly, but first to make changes on another window and
Xthen to call refresh() or wrefresh() to make curscr look like the other
Xwindow. Making a change on curscr results in immediate execution of the
Xchange, this greatly reduces any chances of optimization and therefore results
Xin a sizeable loss of speed (unless you know what you're doing, of course).
X
X   Naming conventions.
X

X   All output and input functions can be used with any window. These functions
Xhave two variants, one to act on the default screen stdscr, and another to
Xact on any window. An example is the function addch(), which adds a character
Xto stdscr. The variant, which adds a character to a window w is called
Xwaddch() and takes w as an argument. This convention is used throughout the
Xpackage. The routines which do not adhere to this convention always take a
Xwindow as an argument.
X   Apart from the w-convention some functions also take a mv- prefix.
XOrdinarily one moves the current coordinates of a window by calling
Xthe move or wmove function. Preceding an I/O function by mv first moves
Xthe current coordinates and then performs the I/O function. In this way
Xone can write the sequence :
X
Xmove(y,x) ;
Xaddch('Q') ;
X
Xas :
X
Xmvaddch(y,x,'Q') ;
X
Xand the sequence :
X
Xwmove(win,y,x) ;
Xwaddch(win,'W') ;
X
Xmay be replaced by :
X
Xmvwaddch(win,y,x,'W') ;
X
XYou may have noticed the following other conventions :
X1. If a window must be specified as a parameter, it is always the first
X   parameter.
X2. Coordinates are specified as (y,x) pairs, where y is in the up/down
X   direction, and x in the letf/right direction, the coordinate (0,0) is
X   in the upper-left corner of the window.
X
X
X   Programming with curses.
X
X   The following sections deal with programming with the library.
XFirst of all curses should be initialized first. This is done by a call to
Xinitscr(). On return curses will have initialized the screen to a state known
Xby curses, and have initialized curscr and stdscr, whereas prior to the
Xinitscr() call, any reference to curscr or stdscr will result in bombs.
XAfter calling initscr() you should perform any terminal status changing
Xroutines like crmode() or nonl(), although these may be performed anytime
Xwhen curses is active. After that you can set up the screen status using
Xfunctions like leaveok() and scrollok() or allocate new windows with newwin()
Xor subwin(), delete them with delwin().
X   Calling initscr() again will either result in a redraw of the screen or
Xanother initialization step when the variables LINES or COLS have changed.
XThis enables you to set the screen size by hand, because normally
Xthe variables LINES and COLS contain the true size of the terminal. Calling

```
Xinitscr ni such a way again, results in the deletion and regeneration of
Xcurscr and stdscr.
X   The basic functions that allow you to change the contents of a window
Xare addch() and move(). Addch() adds a character to a window and move() moves
Xthe current coordinates. Other output functions exist to let you do some
Xmore elaborate things, like addstr() to add an entire string, insertln() to
Xinsert a line etc.. When you have composed the contents of the window to your
Xliking you can call refresh() to make the portion of the terminal screen
Xoverlapped by the window look like the window. When updating one must keep
Xin mind that refresh assumes that parts of the window not changed since the
Xlast update of the window are also not changed on the terminal. Use the
Xtouchwin() function to let curses think the entire window has changed when
Xusing overlapping windows.
X   Calling wrefresh() with curscr as an argument results in a complete redraw
Xof the terminal screen. Because curses has to know what the contents of the
Xphysical screen is at all times, you should not do any other screen output,
Xthan by curses alone. Or if you really have to, do a redraw afterwards. If
Xyou don't, you're bound to end up with some very confused screens.
X   Input is done by the getch() function, which inputs a character from the
SHAR_EOF
echo "End of part 1, continue with part 2"
echo "2" > s2_seq_.tmp
exit 0


From -1630286930377632649
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,53433bdd2da5902e,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-12 13:41:13 PST
Path: utzoo!attcan!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!
gem.mps.ohio-state.edu!apple!sun-barr!newstop!sun!grapevine!
koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i091: curses -- Screen updating package  part02/02
Keywords: shar, C
Message-ID: <34529@grapevine.uucp>
Date: 12 Oct 89 20:41:13 GMT
Sender: news@grapevine.EBay
Reply-To: gray@hplb.hpl.hp.com (Graham Higgins)
Lines: 1062
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: gray@hplb.hpl.hp.com (Graham Higgins)
Posting-number: Volume 2, Issue 91
Archive-name: curses/part02

#!/bin/sh
# this is part 2 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file curses.doc continued
#
```

```
CurArch=2
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> curses.doc
Xkeyboard and echoes it through addch() if echo is set. When using curses
Xyou may notice that, while performing output and other functions, the cursor
Xis switched off, when curses waits for input however the cursor is switched
Xon. Input can be done in either one of three modes : cooked, cbreak and raw.
XThese input modes are primarily introduced to maintain UN*X-compatability, but
Xthey do not work the same, so when porting UN*X-programs special care should
Xbe taken with any input routines. Moreover, you should not perform any input
Xthrough the stdio library, as they will surely echo something on the screen.
XUN*X-curses can handle this because it explicitly sets the echo of the terminal
Xoff, whereas Atari-curses only simulates this behaviour. So : any input only
Xthrough curses.
X   Cooked mode is supposed to be a line-oriented input interface. Using UN*X
Xand curses, cooked input can only be performed, when echo is not set, if it
Xis set cooked mode input turns to cbreak mode for the duration of the
Xinput call. Using Atari and curses, the same restrictions apply, with the
Xexception that the following characters have special meanings :
X
X+---------------+-------+-----------------------------------------+
X| character     | ASCII | action                                  |
X+===============+=======+=========================================+
X| control-C     |   $03 | do a Pterm(-1), i.e. exit immediately   |
X| control-H/BS  |   $08 | do a backspace, not beyond the beginning |
X|               |       | of the input-field.                     |
X| control-U     |   $13 | kill the entire input field             |
X+---------------+-------+-----------------------------------------+
X
X   Cbreak mode is supposed to be a single character oriented interface, with
Xwhich one can receive any character and have no special meaning attached to it.
XThe only exceptions are job-control characters like control-C ( interrupt ),
Xdelete ( kill the process ), control-Z ( suspend the process ) etc. On Atari
Xthis is translated in the following : all characters are received and only
Xcontrol-C has special meaning, which is : terminate the process.
X   Raw mode is also a single character interface and has no special processing
Xwhatsoever. The same holds for Atari.
X   Apart from the above-mentioned functions one should also note that
Xgetch() does not return a character, but a 32-bit value. For UN*X getch()
Xreturns an int, with ( apart from errors ) only the lower 7 or 8 bits set.
XOn Atari the same holds true for cooked mode, in cbreak mode the ASCII value
Xis returned in bits 0 to 7 and the scancode of the key in bits 16 to 23, raw
Xmode adds the keyboard shift state in bits 24 to 30. When entering a string
Xthrough getstr() or wgetstr() only the character portion of the long values
Xare retained.
```

X   Cleaning up is performed by the endwin() function. It releases the storage
Xacquired for curscr and stdscr, switches the terminal to its normal state
Xand puts the cursor in a reasonably comfortable place, i.e. at the bottom
Xof the screen. Note that any windows acquired through newwin() and subwin()
Xstill exist. They should be released from storage prior to calling endwin().
X
X
X   Constants, types and variables defined by curses.h.
X
X   The include file curses.h makes several constants, types and variables
Xavailable to the programmer. In this section something will be told about
Xtheir function and use.
X   The most important thing defined in curses.h is the data type WINDOW. This
Xdatatype is used by curses to maintain data for all windows. It has the
Xfollowing definition :
X
X#define   WINDOW   struct _win_st
X
Xstruct _win_st {
X   WORD   _cury, _curx ;
X   WORD   _maxy, _maxx ;
X   WORD   _begy, _begx ;
X   WORD   _flags ;
X   bool   _clear ;
X   bool   _leave ;
X   bool   _scroll ;
X   WORD   **_y ;
X   WORD   *_firstch ;
X   WORD   *_lastch ;
X   } ;
X
XThe internals of this structure are not normally to be accessed by the user.
XThis structure differs in three aspects from the UN*X-curses WINDOW structure.
XFirst of all the _y field is char in UN*X-curses, WORD in Atari-curses. This
Xhad been done to ensure that all characters in the Atari character set,
Xincluding all diacritical marks, can be displayed, whereas UN*X-curses will
Xonly display characters $20-$7f, or standard ASCII characters. As a matter
Xof act the Atari will display codes $00-$ff, with the exception of tabs,
Xlinefeeds and carriage returns. These have a special meaning.
X Each character can be in standout mode, if so desired. The other changes are
Xin the _firstch, _lastch fields. This reflects the updating algorithms used.
X   The fields have the following meaning : _cury and _curx are the current
Xcoordinates of the window, relative to the window's home, _begy and _begx. This
Xbeginning is relative to the terminals home position (0,0). _maxy and _maxx are
Xthe maximum values allowed for _cury and _curx. _flags is a bit-pattern
Xof several flags describing the structure of the window. _flags can have
Xseveral values OR'ed into it. _SUBWIN means that the window is a subwindow,
Xwhich indicates to delwin() that the character space cannot be freed. The
X_FULLWIN flag indicates that the window is a screen. _STANDOUT means that any
Xadded character is in standout-mode. UN*X-curses also defines the _ENDLINE
Xand _SCROLLWIN flags, they are defined in curses.h for compatability purposes,
Xbut Atari-curses does nothing with these values. The _clear field in the
XWINDOW structure is set when the next refresh on the window should generate

Xa clear screen sequence. Setting the clear flag for curscr will always
Xgenerate a clear screen sequence on any refresh. The _leave field is TRUE
Xif the windows current coordinates are to be set after the last character
Xchanged on the terminal after a refresh. The _scroll field is TRUE, if
Xscrolling is allowed. The _y field is a pointer to an array of pointers to
Xarrays of character values. These values are the actual contents of a window.
XThe lower 8 bits of a character value contain the real character value, the
Xupper 8 bits are used by curses internally. The _firstch and _lastch fields
Xare used by the updating algorithms of curses. Unless you know what you are
Xdoing you are not supposed to tamper with any of these values, curses supplies
Xa full set of functions to set any values which you want to be set.
X   After initializing curses you can be sure the following variables have been
Xset and initialized to their proper values. Some variables may even be set
Xprior to initializing curses. This is marked by a & in the table.
X
X+----------------+------------+---+
X| variable name | type | description |
X+================+============+===+
X| curscr | WINDOW * | curses idea of what the terminal screen looks |
X| | | like. |
X| stdscr | WINDOW * | Default screen for updates. |
X| Def_term | char * | Default terminal type. UN*X-compatability. |
X| My_term   & | bool | UN*X-compatability, does nothing. |
X| ttytype | char * | Full name of the terminal |
X| LINES   & | int | Number of ines curses thinks the terminal has |
X| | | normally 25, but may be changed, see text |
X| COLS   & | int | Number of columns curses thinks the terminal |
X| | | has, may be changed, see text. |
X| ERR | int | If this is set, curses had an internal error |
X| OK | int | If this is set, everything is gone right |
X+----------------+------------+---+
X
X   Also the following constants and macros are defined in curses.h :
X
X+----------------+---+
X| name | description |
X+================+===+
X| reg | storage class register, i.e. register int i => reg int i |
X| bool | boolean type, values can be TRUE or FALSE |
X| TRUE | boolean 'true' flag. |
X| FALSE | boolean 'false' flag. |
X+----------------+---+
X
X   Another difference between Atari-curses and UN*X-curses is that UN*X-curses
Xreads in a lot of variables described in /etc/termcap, no such thing exists
Xfor Atari, and is not needed anyway. However these variables are available
Xto the programmer in UN*X-curses, but not in Atari-curses. These include
Xthings like strings, which when output to the terminal cause some action to be
Xtaken, like the string VB, for a visible bell, or DM, enter delete mode. Care
Xshould be taken with UN*X-programs that perform such actions.
X
X
X   Description of individual functions.

```
X
X   The following is a description of all the individual routines, a lot of them
Xare defined as macros, notably those that take stdscr as the window and
Xfunctions that can be preceded by "mv". If such is the case, it is stated by
Xa '!' a little further on the line.
X
X   Output functions.
X
X
Xaddch(c)                   !
Xchar c ;
X
Xmvaddch(y,x,c)             !
Xint y,x ;
Xchar c ;
X
Xmvwaddch(win,y,x,c)        !
XWINDOW *win ;
Xint y,x ;
Xchar c ;
X
Xwaddch(win,c)
XWINDOW *win ;
Xchar c ;
X
X   Add the character c at the current coordinates of the window w. The
Xfollowing characters cause special processing : a newline ('\n') causes
Xthe line to be cleared to the end, and the current coordinates to be changed to
Xthe beginning of the next line if newline mapping is on, or to the next line
Xat the same column if newline mapping isn't on. A return ('\r') will move the
Xcursor to the beginning of the line on the window. Tabs ('\t') will be
Xexpanded into spaces into the normal tabstop positions of every eight
Xcharacters.
X
Xaddstr(s)                  !
Xchar *s ;
X
Xmvaddstr(y,x,s)            !
Xint y,x ;
Xchar *s ;
X
Xmvwaddstr(win,y,x,s)       !
XWINDOW *win ;
Xint y,x ;
Xchar *s ;
X
Xwaddstr(win,s)
XWINDOW *win ;
Xchar *s ;
X
X   Add the string s to the window at the current coordinates. Addstr() calls
Xaddch repeatedly to add the characters.
X
```

```
Xbox(win,vert,hor)
XWINDOW *win ;
Xchar vert, hor ;
X
Xmvbox(win,y,x,vert,hor)   !
XWINDOW *win ;
Xint y, x ;
Xchar vert, hor ;
X
X   Draws a box around the window using vert as the vertical drawing character
Xand hort as the horizontal drawing character.
X
Xclear()                   !
X
Xwclear(win)
XWINDOW *win ;
X
X   Reset the window to blanks. If win is a screen this will cause a clear
Xscreen sequence to be sent on the next refresh. The current coordinates
Xare also moved to the home-position (0, 0).
X
Xclearok(scr, boolf)       !
XWINDOW *scr ;
Xbool boolf
X
X   Sets the clear-flag for the screen scr to boolf. If boolf is TRUE this
Xwill generate a clear screen sequence on the next refresh. If it is FALSE
Xrefresh may be stopped from generating a clear screen sequence.
X
Xclrtobot()                !
X
Xwclrtobot(win)
XWINDOW *win ;
X
X   Clear the window from the current coordinates to the bottom. The current
Xcoordinates aren't changed.
X
Xclrtoeol()                !
X
Xwclrtoeol(win)
XWINDOW *win ;
X
X   Clear the window from the current coordinates to the end of the line. This
Xdoesn't change the current coordinates.
X
Xdelch()                   !
X
Xmvdelch(y,x)              !
Xint y, x ;
X
Xmvwdelch(win,y,x)         !
XWINDOW *win ;
Xint y, x ;
```

```
X
Xwdelch(win)
XWINDOW *win ;
X
X   Delete the character at the current coordinates. Characters on the same
Xline after the deleted character shift to the left and the last character
Xbecomes blank.
X
Xdeleteln()               !
X
Xmvdeleteln(y,x)          !
Xint y, x ;
X
Xmvwdeleteln(win,y,x)     !
XWINDOW *win ;
Xint y,x ;
X
Xwdeleteln(win)
XWINDOW *win ;
X
X   Delete the current line, move every line under it one line up and make the
Xbottom line blank. This doesn't change the current coordinates.
X
Xerase()                  !
X
Xwerase(win)
XWINDOW *win ;
X
X   Sets the entire window to blanks without setting the clear flag. See
Xclear(). The difference is that erase doesn't cause a clear screen sequence
Xto be generated.
X
Xinsch(c)                 !
Xchar c ;
X
Xmvinsch(y,x,c)           !
Xint y, x ;
Xchar c ;
X
Xmvwinsch(win,y,x,c)      !
XWINDOW *win ;
Xint y, x ;
Xchar c ;
X
X   Insert the character c at the current coordinates. each character after
Xthe current character shifts one place to the right and the rightmost character
Xfalls off.
X
Xinsertln()               !
X
Xmvinsertln(y,x)          !
Xint y, x ;
X
```

```
Xmvwinsertln(win,y,x)      !
XWINDOW *win ;
Xint y, x ;
X
X   Insert a line above the current line. Every line below will scroll down, and
Xthe bottom line will disappear. The inserted line will be blank. The current
Xcoordinates aren't chnaged by this function.
X
Xmove(y,x)                 !
Xint y, x ;
X
Xwmove(win,y,x)
XWINDOW *win ;
Xint y, x ;
X
X   Change the current coordinates of window win to (y, x). Whenever one of the
Xother output or input functions are preceded by "mv", this function is
Xexecuted first.
X
Xoverlay(win1,win2)
XWINDOW *win1, *win2 ;
X
X   The contents of win1 are placed on win2 at their respective starting
Xcoordinates. Those contents of win1, which would not fit in win2, are
Xdiscarded. Blanks on win1 leave the contents of win2 untouched.
X
Xoverwrite(win1, win2)
XWINDOW *win1, *win2 ;
X
X   The contents of win1 are placed on win2 at their respective starting
Xcoordinates. Those contents of win1, which would not fit in win2, are
Xdiscarded.
X
Xrefresh()                  !
X
Xwrefresh(win)
XWINDOW *win ;
X
X   Update the terminal screen with the contents of the window. Only those parts
Xcovered by the window will be updated. If the window is a screen and its
Xclear flag has been set, then a clear screen sequence will be generated.
X
Xmvstandend(y,x)            !
Xint y, x ;
X
Xmvwstandend(win,y,x)       !
XWINDOW *win ;
Xint y, x ;
X
Xstandend()                 !
X
Xwstandend(win)
XWINDOW *win ;
```

```
X
X   Stop putting characters onto win in standout mode.
X
Xmvstandout(y,x)           !
Xint y, x ;
X
Xmvwstandout(win,y,x)      !
XWINDOW *win ;
Xint y, x ;
X
Xstandout()                !
X
Xwstandout(win)
XWINDOW *win ;
X
X   Start putting characters onto win in standout mode, i.e. display inverted
Xcharacters.
X
X
X   Input functions.
X
X
Xcrmode()
X
X   Set the terminal in cbreak mode.
X
Xnocrmode()
X
X   Reset the terminal from cbreak mode into cooked mode.
X
Xecho()
X
X   Let the terminal echo characters on input.
X
Xnoecho()
X
X   Do not allow the terminal to echo characters on input.
X
Xgetch()                    !
X
Xmvgetch(y,x)               !
Xint y,x ;
X
Xmvwgetch(win,y,x)          !
XWINDOW *win ;
Xint y, x ;
X
Xwgetch(win)
XWINDOW *win ;
X
X   Get a character from the terminal and echo it on the window if echo is set.
XSee the section on programming with curses for more details about input modes.
XRemember getch() returns a 32-bit value, and not a char.
```

```
X
Xgetstr(s)                !
Xchar *s ;
X
Xmvgetstr(y,x,s)          !
Xint y,x ;
Xchar *s ;
X
Xmvwgetstr(win,y,x,s)     !
XWINDOW *win ;
Xint y, x ;
Xchar *s ;
X
Xwgetstr(win,s)
XWINDOW *win ;
Xchar *s ;
X
X   Get a string through the window win and put it in the string s, which is
Xassumed to be large enough to hold the data. To get the input it calls
Xgetch() repeatedly. The end of the input is designated by any of the
Xfollowing characters, a carriage return, a linefeed, a EOF or control-D, and
Xany character, which results in a '\0'. This character is stripped of the
Xstring and replaced by a null character to indicate the end of the string.
X
Xraw()
X
X   Set the terminal in raw input mode.
X
Xnoraw()
X
X   Set the terminal from raw into cooked input mode.
X
X
X   Miscellaneous functions.
X
X
Xdelwin(win)
XWINDOW *win ;
X
X   Deletes the window. All resources ( i.e. storage ) held by the window is
Xfreed for future use. As subwindows share their character space with another
Xwindow their character space is not freed. To delete a window with
Xsubwindows, you should first delete the subwindows and then the root window,
Xas any access to a subwindow after deletion of its parent window is a sure
Xroute to disaster.
X
Xendwin()
X
X   Clean up after curses. You should call this prior to termination of
Xyour program.
X
Xgetyx(win,y,x)            !
XWINDOW *win ;
```

```
Xint y,x ;
X
X   This macro puts the window current coordinates into the variables y and x.
XNote that you do not supply the address of the variables, as it is a macro.
X
Xinch()                    !
X
Xmvinch(y,x)               !
Xint y,x ;
X
Xmvwinch(w,y,x)            !
XWINDOW *w ;
Xint y, x ;
X
Xwinch(win)                !
XWINDOW *win ;
X
X   This macro returns the value of the character at the current coordinates
Xof the window. This does not change the contents of the window. The functions
Xmvinch() and mvwinch() are not supported by standard UN*X-curses, although
Xthey probably are by most UN*X-implementations.
X
Xinitscr()
X
X   Initialize curses. You should call this before any of the other calls.
X
Xleaveok(win,flag)
XWINDOW *win ;
Xbool   flag ;
X
X   Sets the _leave field of the window win to the value flag. If flag is
XTRUE then after an update on the terminal, the window's current
Xcoordinates will be set to the coordinates immediately after the last updated
Xcharacter. If flag is FALSE, then the cursor will be moved to the windows
Xcurrent coordinates. This flag is FALSE by default.
X
Xlongname(termbuf,name)
Xchar *termbuf, *name ;
X
X   Introduced for UN*X-compatability, it returns the contents of ttytype
Xin name, regardless of the contents of termbuf. This is not really consistent
Xwith UN*X-curses.
X
Xmvwin(win,y,x)
XWINDOW *win ;
Xint y, x ;
X
X   Move the home position of the window win, from its current coordinates to
Xthe coordinates specified in y and x. You cannot move a window off the
Xterminal screen, not even by a character.
X
XWINDOW *newwin(l,c,by,bx)
Xint l, c, by, bx ;
```

```
X
X   Create a new window with l lines and c columns starting at position (by,bx).
XIf either l or c is zero, then the dimension will be set to ( LINES - by ),
X( LINES - bx ). To get a screen, one would say newwin(0,0,0,0).
X
Xnl()
X
X   Set curses to newline mapping. This has consequences for adding the
Xcharacter '\n'. See addch() for these consequences.
X
Xnonl()
X
X   Set curses to do no newline mapping. This is the default. In UN*X-curses
Xyou get the fastest updates when you have no newline mapping, Atari-curses
Xdoesn't care about it.
X
Xscrollok(win,flag)
X
X   Set the _scroll field of the window win to flag. If flag is FALSE, the
Xterminal is not allowed to scroll, which is also the default.
X
Xtouchwin(win)
XWINDOW *win ;
X
X   Force curses to believe that the entire window has changed on the next
Xrefresh. This is useful for overlapping windows.
X
XWINDOW *subwin(win,l,c,by,bx)
X
X   Create a new window with a shared character space,as opposed to newwin(),
Xwhich allocates a new character space. See newwin() for more details about
Xthe l, c, by and bx parameters. The new window will share the character space
Xwith the window win. The subwindow must be smaller or the same size as
Xthe parent window. Changes on either window are reflected on the other window.
XMoving subwindows with mvwin() will create some pretty weird results, because
Xthe mapping of the subwindow to the parent window will tend to get displaced.
XFor instance, if I have a one character subwindow of a screen at coordinates
X(2,3) and if I move the subwindow to say (4,5), then if I change the character
Xon the subwindow at (4,5), the parent window will have changed at (2,3).
XI don't know whether this is UN*X-compatible, so take care here.
X
X
X   Detail functions.
X
X
X   These routines are at the core of UN*X-curses, and can be used on a
Xstandalone basis. Atari-curses has a very limited implemntation of this, so
Xif a program uses these, you'd better rewrite the whole thing. Most likely,
Xthe results of using these won't be the same on either of the two
Ximplementations.
X
Xmvcur(lasty,lastx,newy,newx)
Xint lasty, lastx, newy, newx ;
```

```
X
X   Move the terminals cursor from (lasty,lastx) to (newy,newx). Don't use it
Xwhen you use any of the screen routines.
X
Xscroll(win)
XWINDOW *win ;
X
X   Scroll the window upward one line. You don't normally want to do this.
X
X
X   Functions not supported by Atari-curses.
X
X
X   Several functions are not supported by Atari-curses, but are by UN*X-curses.
XThe names of these functions are given, as well as a reason for not
Ximplementing them.
X
Xmvprintw(), mvwprintw(), printw() and wprintw()
X
X   Performs a printf() on a window. I didn't want to implement this for two
Xreasons :
X1. This introduces compiler dependant code.
X2. A large overhead in coding is required. ( I'm very lazy. )
XThis omission can easily be gotten around with by inline replacement with :
X                 sprintf(a_string, format, ... ) ;
X                 addstr(a_string) ;
X
Xmvscanw(), mvwscanw(), scanw() and wscanw()
X
X  Performs a scanf() on a window. The same reasons apply here as the reasons
Xfor printw(). Replace it inline with :
X                 getstr(a_string) ;
X                 sscanf(a_string, format, ... ) ;
X
Xuncntrl()
X
X  Returns a string which is a representation of a character, nice debugging
Xfeature for UN*X-curses, but pretty meaningless for Atari-curses as
XAtari-curses can show any character.
X
Xgettmode()
X
X   A function called by UN*X-initscr(), it gets tty stats, which is
XUN*X-dependant anyway.
X
Xsavetty() and resetty()
X
X   Save and reset tty characteristics, used by UN*X-curses to initialize and
Xclean up respectively. Not necessary on Atari.
X
Xsetterm(name)
X
X   Set terminal characteristics to those of the terminal with name name.
```

```
XPretty useless on Atari.
X
Xtstp()
X
X   Function which is used in UN*X-curses to save the current state when a
Xprocess is put to sleep and to restore it when it is restarted. This is the
Xnormal catch for the signal SIGTSTP, which Atari users won't be able to receive
Xanyway.
X
X
X   Caveats and bugs.
X
X
X   There are differences between Atari-curses and UN*X-curses, especially in
Xthe area of input and the more detailed workings of curses. I will
Xprobably not change the workings of any input routines, because if I did
Xapplication programs wouldn't have any control over say function keys, cursor
Xkeys, etc. Such keys really make a difference in user friendly programs, so
XI'd better let a programmer have access to them. The more detailed workings
Xof curses are best left out too, as they concern themselves with multiple types
Xof terminals, where the Atari-curses is only fitted to a single
Xterminaltype, i.e. the Atari's VT52 emulation mode. Another problem
Xis possibly the low-resolution mode of the Atari, I don't have access to
Xa color monitor, so I wasn't able to test it in either medium or low-res.
XMedium probably works fine, but low-res ?. So, if you don't do anything too
Xfancy, you should be allright. One more thing : when you write portable
Xprograms, keep in mind that the Atari-curses is a lot faster than any
Ximplementation, which sends data over a terminal line. Comparing speeds
Xwill put any program on the Atari seemingly on a 50+ kilobaud line.
XAlso beware of funny characters, i.e. portable programs require character
Xranges between $20 and $7F, Atari-curses can and will handle a lot more. This
Xwarning should especially be heeded by say Germans and French, diacritical
Xmarks may screw things up real bad.
SHAR_EOF
chmod 0640 curses.doc || echo "restore of curses.doc fails"
sed 's/^X//' << 'SHAR_EOF' > curses.h &&
X/* ************************************************************************ */
X/*                                                                          */
X/*   CURSES.H                      include file for programs using CURSES   */
X/*                                                                          */
X/* ************************************************************************ */
X/*                                          */
X/* This source and resulting object may be modified, used or distributed by */
X/* anyone who so desires under the following conditions :                   */
X/*                                          */
X/*  1) This notice and the copyright notice shall not be removed or     */
X/*     changed.                                    */
X/*  2) No credit shall be taken for this source and resulting objects   */
X/*  3) This source or resulting objects is not to be traded, sold or    */
X/*     used for personal gain or profit.                   */
X/*  4) Inclusion of this source or resulting objects in commercially    */
X/*     available packages is forbidden without written consent of the   */
X/*     author of this source.                          */
```

```
X/*                                                                          */
X/* ************************************************************************ */
X
X#include "portab.h"
X#include "stdio.h"
X
X#define reg           register
X#define bool      BYTE
X#define TRUE      1
X#define FALSE         0
X
X#define _SUBWIN       01
X#define _ENDLINE  02
X#define _FULLWIN  04
X#define  _SCROLLWIN    010
X#define  _STANDOUT0200
X#define  WINDOW        struct _win_st
X
X#define TOUCHED       0x0200
X#define STANDOUT  0x0100
X
X#define addch(c)           waddch(stdscr, c)
X#define mvaddch(y,x,c)     { wmove(stdscr,y,x) ; waddch(stdscr,c) ; }
X#define mvwaddch(w,y,x,c)  { wmove(w,y,x) ; waddch(w,c) ; }
X#define addstr(s)          waddstr(stdscr,s)
X#define mvaddstr(y,x,s)         { wmove(stdscr,y,x) ; waddstr(stdscr,s) ; }
X#define mvwaddstr(w,y,x,s) { wmove(w,y,x) ; waddstr(w,s) ; }
X#define clear()                 wclear(stdscr)
X#define clearok(w,f)       { w->_clear = (w->_flags & _FULLWIN) ? f : w-
>_clear ; }
X#define clrtobot()              wclrtobot(stdscr)
X#define clrtoeol()              wclrtoeol(stdscr)
X#define delch()                 wdelch(stdscr)
X#define mvdelch(y,x)       { wmove(stdscr,y,x) ; wdelch(stdscr) ; }
X#define mvwdelch(w,y,x)         { wmove(w,y,x) ; wdelch(w) ; }
X#define deleteln()              wdeleteln(stdscr)
X#define mvdeleteln(y,x)         { wmove(stdscr,y,x) ; wdeleteln(stdscr) ; }
X#define mvwdeleteln(w,y,x) { wmove(w,y,x) ; wdeleteln(w) ; }
X#define erase()                 werase(stdscr)
X#define insch(c)           winsch(stdscr,c)
X#define mvinsch(y,x,c)     { wmove(stdscr,y,x) ; winsch(stdscr,c) ; }
X#define mvwinsch(w,y,x,c)  { wmove(w,y,x) ; winsch(w,c) ; }
X#define insertln()              winsertln(stdscr)
X#define mvinsertln(y,x)         { wmove(stdscr,y,x) ; winsertln(stdscr) ; }
X#define mvwinsertln(w,y,x) { wmove(w,y,x) ; winsertln(w) ; }
X#define move(y,x)          wmove(stdscr,y,x)
X#define refresh()          wrefresh(stdscr)
X#define standout()              wstandout(stdscr)
X#define standend()              wstandend(stdscr)
X#define getch()                 wgetch(stdscr)
X#define mvgetch(y,x)       ( wmove(stdscr,y,x) , wgetch(stdscr) )
X#define mvwgetch(w,y,x)         ( wmove(w,y,x) , wgetch(w) )
X#define getstr(s)          wgetstr(stdscr,s)
```

```
X#define mvgetstr(y,x,s)         { wmove(stdscr,y,x) ; wgetstr(stdscr,s) ; }
X#define mvwgetstr(w,y,x,s) { wmove(w,y,x) ; wgetstr(w,s) ; }
X#define getyx(w,y,x)       { y = w->_cury ; x = w->_curx ; }
X#define inch()                  (stdscr->_y[stdscr->_cury][stdscr->_curx])
X#define mvinch(y,x)             ( wmove(stdscr,y,x) , stdscr->_y[stdscr->_cury]
[stdscr->_curx])
X#define mvwinch(w,y,x)     ( wmove(w,y,x) , w4->_y[w->_cury][w->_curx])
X#define winch(w)           (w->_y[w->_cury][w->_curx])
X
Xstruct _win_st {
X    WORD _cury, _curx ;
X    WORD _maxy, _maxx ;
X    WORD _begy, _begx ;
X    WORD _flags ;
X    BYTE _clear ;
X    BYTE _leave ;
X    BYTE _scroll ;
X    WORD **_y ;
X    WORD *_firstch ;
X    WORD *_lastch ;
X} ;
X
Xextern WINDOW *curscr ;
Xextern WINDOW *stdscr ;
Xextern char  *Def_term ;
Xextern bool  My_term ;
Xextern char  *ttytype ;
Xextern int   LINES ;
Xextern int   COLS ;
Xextern int   ERR ;
Xextern int   OK ;
SHAR_EOF
chmod 0640 curses.h || echo "restore of curses.h fails"
sed 's/^X//' << 'SHAR_EOF' > curses.man &&
XCURSES(3X)                                                          CURSES(3X)
X
X
XNAME
X
X    curses - screen functions with "optimal" cursor motion
X
X
XDESCRIPTION
X
X   The functions in the curses-package give you an implementation of the
XUN*X-curses package for the Atari ST series computer. It allows you to do
Xterminal-oriented graphics of a crude nature. An image is kept of what the
Xcurrent screen looks like, while the user creates a new image. Refresh()
Xmust be called to make the current screen look like the one the user has
Xprepared. Initscr() must be called to initialize the package and endwin()
Xto clean up before exiting.
X
XSEE ALSO
```

X
X   Screen Updating and Cursor Movement Optimization : A Library Package,
X   Ken Arnold.
X   The CURSES screen updating and cursor movement package : A collection
X   of C-functions for the Atari ST-series computers, R. van 't Veen.
X
XAUTHORS
X
X   UN*X-curses : Ken Arnold.
X   Atari-curses : Rene van 't Veen.
X
XFUNCTIONS
X
X   addch(c)   add a character to stdscr
X   addstr(s)   add a string to stdscr
X   box(win,v,h)   draw a box around a window
X   crmode()   set cbreak mode
X   clear()   clear stdscr
X   clearok(scr,f)   set clear flag for scr
X   cltobot()   clear to bottom on stdscr
X   clrtoeol()   clear to end of line on stdscr
X   delch()   delete a character
X   deleteln()   delete a line
X   delwin(w)   delete window w
X   echo()   set echo mode
X   endwin()   end window mode
X   erase()   erase stdscr
X   getch()   get a char through stdscr
X   getstr()   get a string through stdscr
X   getyx(w,y,x)   get current coordinates
X   inch()   get char at current coordinates
X   initscr()   initialize curses
X   insch()   insert a character
X   insertln()   insert a line
X   leaveok(w,f)   set leave flag for window w
X   longname(termbuf, name)   get a long terminal name
X   move(y,x)   move to (y,x) on stdscr
X   mvcur(ly,lx,ny,nx)   actually move the cursor
X   newwin(lines,cols,begin_y,begin_x)   create a window
X   nl()   set newline mapping on
X   nocrmode()   unset cbreak mode
X   noecho()   unset echo mode
X   nonl()   set newline mapping off
X   noraw()   unset raw mode
X   overlay(win1,win2)   overlay win1 on win2
X   overwrite(win1,win2)   overwrite win1 on win2
X   raw()   set raw mode
X   refresh()   make terminal look like stdscr
X   scroll(win)   scroll win one line up
X   scrollok(w,f)   set scroll flag for window win
X   standend()   leave standout mode
X   standout()   enter standout mode
X   subwin(win,lines,cols,begin_y,begin_x)   create a sub window

```
X   touchwin(win)                              change everything on win
X   waddch(win,c)                              add char to win
X   waddstr(win,s)                             add string to win
X   wclear(w)                                  clear w
X   wclrtobot(w)                               clear w to bottom of w
X   wclrtoeol(w)                               clear w to end of line
X   wdelch(w)                                  delete character from w
X   wdeleteln(w)                               delete line from w
X   werase(w)                                  erase w
X   wgetch(w)                                  get a character through w
X   wgetstr(w,s)                               get a string through w
X   winch(w)                                   get char at current (y,x) of w
X   winsch(w,c)                                insert a char on w
X   winsertln(w)                               insert a line on w
X   wmove(w,y,x)                               set current coordinates of w
X   wrefresh(w)                                make terminal look like w
X   wstandend(w)                               leave standout mode for w
X   wstandout(w)                               enter standout mode for w
X
XBUGS
X
X   Hopefully none, but you never know ...
X   If you find any, please let me know,
X   I ( Rene ) am at ..!mcvax!nikhefh!u13 .
SHAR_EOF
chmod 0640 curses.man || echo "restore of curses.man fails"
sed 's/^X//' << 'SHAR_EOF' > portab.h &&
X/*
X * This file re-implements portions of the Alcyon portab.h needed
X * by curses. If you have a more complete portab.h that contains
X * the following definitions, use it.
X */
X
X#define  BYTE char
X#define  WORD int
X#define  LONG long
SHAR_EOF
chmod 0640 portab.h || echo "restore of portab.h fails"
sed 's/^X//' << 'SHAR_EOF' > readme.too &&
X        Notes for the Megamax Version of Curses
X
XThis version of curses compiles with the Megamax C compiler. A couple
Xof bug fixes were needed and the wgetstr routine was hacked quite a bit.
X
X
XNotes:
X
X1. The overlay() routine was renamed woverlay() to avoid a conflict with
X   the Megamax reserved word.
X
X2. The entire file was placed in an overlay because I needed the space.
X   You may want to get rid of the overlay line in curses.c depending on
X   what you're linking curses with.
```

```
X
X3. The routines printw() and scanw() are implemented in the usual brute-
X   force way since Megamax doesn't have varargs. These are at the end
X   of curses.c.
X
X4. Remember to change your init.c to get extra stack space.
X
X5. The file portab.h contains the definitions needed from the Alcyon
X   file of the same name. If you have a more complete version, use it.
X
X6. twinkle.tos isn't in this distribution because I wanted to post this
X   stuff as clear text.
X
X
XTony Andrews
Xihnp4!onecom!wldrdg!tony
XWildridge Consulting, Inc.
XBoulder, CO
SHAR_EOF
chmod 0640 readme.too || echo "restore of readme.too fails"
sed 's/^X//' << 'SHAR_EOF' > readme.txt &&
XThis is Atari-curses, read the curses.doc file about installing first.
X
XThe archive should contain the following files :
X
XCURSES.C         Source for CURSES, compile once.
XCURSES.H         Include file for CURSES.C and any program using CURSES.
XCURSES.MAN       CURSES short docs, a manual page for UN*X-freaks.
XCURSES.DOC       CURSES long docs, this is what you will want to read when
X                 you are going to use CURSES.
XREADME.TXT       This file.
XTWINKLE.C        Sample application.
XTWINKLE.TOS      Ready to run sample application.
X
X
XThis is essentially a shareware program, its usage is subject to the following
Xconditions :
X
XPUBLIC-DOMAIN       : don't take the credit for it.
XCommercial purposes : Send 5 US-dollars or equivalent currency and return
X                      postage to me to get a written consent to use it.
X
XCURSES may be distributed to anyone, provided that all of the above files
Xremain in the archive, and the various copyright notices remain intact.
X
XBug reports and suggestions are welcome, send me e-mail.
X
XRene van't Veen                  ( ...!mcvax!nikhefh!u13 )
XDebussystraat 27
X1817 GL Alkmaar
XThe Netherlands
SHAR_EOF
chmod 0640 readme.txt || echo "restore of readme.txt fails"
```

```
sed 's/^X//' << 'SHAR_EOF' > twinkle.c &&
X/* sample application using curses calls, this takes about 25 seconds on the */
X/* Atari and over 6 minutes on a mainframe, using a vt100 on a 4800 baud     */
X/* line. Even on the mainframe it uses 24 seconds of computation time.       */
X
X#include "portab.h"
X#include "curses.h"
X
X#define NCOLS 80
X#define NLINES 24
X#define MAXPATT 4
X
Xstruct locs {
X    char y,x ;
X    } ;
X
Xtypedef struct locs LOCS ;
X
XLOCS Layout[NCOLS*NLINES] ;
X
Xint pat ;
Xint nstar ;
X
Xmain()
X{
X    int i ;
X
X    initscr() ;
X    leaveok(stdscr,1) ;
X    for ( i = 0 ; i < 10 ; i++ )
X         {
X         makeboard() ;
X         puton('*') ;
X         puton(' ') ;
X         } ;
X    endwin() ;
X}
X
Xmakeboard()
X{
X    int y,x ;
X    LOCS *lp ;
X
X    pat = rand() % MAXPATT ;
X    lp = Layout ;
X    for ( y = 0 ; y < NLINES ; y++ )
X         for ( x = 0 ; x < NCOLS ; x++ )
X              if ( ison(y,x) )
X                   {
X                   lp->y = y ;
X                   lp->x = x ;
X                   lp++ ;
X                   } ;
```

```
X    nstar = lp - Layout ;
X}
X
Xison(y,x)
Xint y,x ;
X{
X    switch ( pat )
X         {
X         case 0:
X              return(!(y & 0x01)) ;
X         case 1:
X              if ( x >= NLINES && y >= NCOLS )
X                   return(0) ;
X              if ( y < 3 || y >= NLINES - 3 )
X                   return(1) ;
X              return((x < 3 || x >= NCOLS - 3 )) ;
X         case 2:
X              return(((x+y) & 0x01)) ;
X         case 3:
X              return(( y>= 9 && y <= 15 )) ;
X         } ;
X}
X
Xputon(ch)
Xchar ch ;
X{
X    LOCS  *lp, *end, temp ;
X    int r ;
X
X    end = &(Layout[nstar]) ;
X    for ( lp = Layout ; lp < end ; lp++ )
X         {
X         r = ((unsigned int) rand()) % nstar ;
X         temp = *lp ;
X         *lp = Layout[r] ;
X         Layout[r] = temp ;
X         } ;
X    for ( lp = Layout ; lp < end ; lp++ )
X         {
X         mvaddch(lp->y,lp->x,ch) ;
X         refresh() ;
X         } ;
X}
SHAR_EOF
chmod 0640 twinkle.c || echo "restore of twinkle.c fails"
rm -f s2_seq_.tmp
echo "You have unpacked the last part"
exit 0


From -5102280164041886575
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,64c53fbd6bfbb5b9,start
```

X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-24 18:47:19 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!ames!apple!sun-barr!newstop!sun!
grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i092: sozobon1.2 -- Update to Sozobon C compiler part01/09
Keywords: shar, C
Message-ID: <34603@grapevine.uucp>
Date: 25 Oct 89 01:47:19 GMT
Sender: news@grapevine.EBay
Reply-To: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Lines: 2120
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Posting-number: Volume 2, Issue 92
Archive-name: sozobon1.2/part01

     Here is part 1 of the Sozobon 1.2 release. This is an intermediate
release that only includes the compiler "hcc.ttp" and the optimizer "top.ttp".
The compiler has bug fixes and some minor preprocessor enhancements. The
optimizer is improved with a "registerizing" pass.

     Johann Ruegg
     ncar!dunike!onecom!wldrdg!hans

```
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of archive 1 (of 9)."
# Contents:  hcc hcc/BSTOK.H hcc/COOKIE.H hcc/FLAGS.H hcc/GEN.H
#   hcc/MAKEFILE hcc/OPS.H hcc/PARAM.H hcc/STTOK.H hcc/SUBS_TOS.S
#   hcc/TOK.H hcc/TOS.LD hcc/TYTOK.H top top/HEALTH.C top/INST.H
#   top/IO.C top/MAIN.C top/MAKEFILE top/OPCODES.C top/OPCODES.H
#   top/SYM.C top/TOP.H top/UTIL.C
# Wrapped by koreth@panarthea on Tue Oct 24 18:40:44 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test ! -d 'hcc' ; then
    echo shar: Creating directory \"'hcc'\"
    mkdir 'hcc'
fi
if test -f 'hcc/BSTOK.H' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/BSTOK.H'\"
else
echo shar: Extracting \"'hcc/BSTOK.H'\" \(1464 characters\)
sed "s/^X//" >'hcc/BSTOK.H' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X *
X * Permission is granted to anyone to use this software for any purpose
```

```
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  bstok.h
X *
X *  basic token values
X *  (some compilers cant do enums, so use defines)
X */
X
X#ifndef ENUMS
X#define BADTOK    0
X#define ID   1
X#define ICON 2
X#define FCON 3
X#define SCON 4
X#define DPARAM    6    /* parameter to #define */
X#define EOFTOK    7
X#define POSTINC 8 /* DOUBLE '+' is pre-inc */
X#define POSTDEC 9 /* DOUBLE '-' is pre-dec */
X#define TCONV     10   /* coertion */
X#define TSIZEOF 11     /* sizeof(type expr) */
X#define NL   12
X#define WS   13   /* white space */
X#define SCON2     14   /* <string> */
X#define PTRDIFF 0x18
X#define PTRADD    0x19
X#define PTRSUB    0x1a
X#define ARROW     0x1b
X#define LTEQ 0x1c
X#define GTEQ 0x1d
X#define NOTEQ     0x1e
X#else
Xenum {
X    BADTOK, ID, ICON, FCON, SCON, DPARAM, EOFTOK,
X    POSTINC, POSTDEC, TCONV, TSIZEOF, NL, WS, SCON2,
X    PTRDIFF, PTRADD, PTRSUB, ARROW, LTEQ, GTEQ, NOTEQ
X};
X#endif
X
X#define ASSIGN    0x100+    /* add to char of op */
X#define DOUBLE    0x80+
X#define UNARY     0x200+
X
X#define STAR (UNARY '*')   /* used a lot */
X
X#define isassign(x)    (x >= 0x100)
END_OF_FILE
if test 1464 -ne `wc -c <'hcc/BSTOK.H'`; then
    echo shar: \"'hcc/BSTOK.H'\" unpacked with wrong size!
```

```
fi
# end of 'hcc/BSTOK.H'
fi
if test -f 'hcc/COOKIE.H' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/COOKIE.H'\"
else
echo shar: Extracting \"'hcc/COOKIE.H'\" \(1088 characters\)
sed "s/^X//" >'hcc/COOKIE.H' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X * 
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  cookie.h
X */
X
X/* restrictions on type */
X
X#define R_INTEGRAL    1
X#define R_FLOATING    2
X#define R_POINTER 4
X#define R_STRUCT  8
X#define R_ARITH       (R_INTEGRAL|R_FLOATING)
X#define R_SCALAR  (R_ARITH|R_POINTER)
X#define R_ASSN        (R_SCALAR|R_STRUCT)
X
X/* cookies */
X#define FORSIDE       1
X#define FORPUSH       2
X#define FORCC         3
X#define FORIMA        4
X#define FORADR        5
X#define FORINIT       6
X#define IND0     7
X#define RETSTRU       8
X
X/* eval flags derived from cookie */
X#define NOVAL_OK  1   /* value is optional */
X#define CC_OK         2   /* just setting cond. codes okay */
X#define IMMA_OK       4   /* immediate OREG is okay */
END_OF_FILE
if test 1088 -ne `wc -c <'hcc/COOKIE.H'`; then
    echo shar: \"'hcc/COOKIE.H'\" unpacked with wrong size!
fi
# end of 'hcc/COOKIE.H'
fi
if test -f 'hcc/FLAGS.H' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/FLAGS.H'\"
```

```
else
echo shar: Extracting \"'hcc/FLAGS.H'\" \(1138 characters\)
sed "s/^X//" >'hcc/FLAGS.H' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  flags.h
X *
X *  token or e_flags flags
X */
X
X#define CAN_U     2    /* can be unary op */
X#define CAN_AS    4    /* can be op-assign */
X#define SPECIAL 8 /* special token beginner ("'/*) */
X#define SEE_U     0x10 /* for ICON */
X#define SEE_L     0x20 /* for ICON */
X#define C_NOT_A 0x40  /* commutative but not associative */
X#define C_AND_A 0x80  /* commutative and associative */
X#define IMMEDID   0x100    /* Ptr to XXX ID which is really
X                  Array of XXX or Fun ret XXX */
X#define CHILDNM   0x200    /* my name is my (left) child's name */
X#define SIDE_CC 0x400 /* cc set as side effect of op. */
X#define RCHILDNM 0x800/* my name is my (right) child's name */
END_OF_FILE
if test 1138 -ne `wc -c <'hcc/FLAGS.H'`; then
    echo shar: \"'hcc/FLAGS.H'\" unpacked with wrong size!
fi
# end of 'hcc/FLAGS.H'
fi
if test -f 'hcc/GEN.H' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/GEN.H'\"
else
echo shar: Extracting \"'hcc/GEN.H'\" \(1400 characters\)
sed "s/^X//" >'hcc/GEN.H' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  gen.h
X */
```

```
X
X#include "cookie.h"
X
X/* parameters for case */
X#define C_SIMPLE  8    /* use simple if n <= this */
X#define C_RATIO       3    /* use table if max-min/n <= this */
X
X/* branch types */
X/* designed so that pairs (i,i+1) with odd i represent opposites */
X#define B_EQ 1
X#define B_NE 2
X#define B_LT 3
X#define B_GE 4
X#define B_LE 5
X#define B_GT 6
X#define B_YES     7
X#define B_NO 8
X#define B_ULT     9
X#define B_UGE     10
X#define B_ULE     11
X#define B_UGT     12
X
X/* derived nodes */
X/* re-use stmt keywords for now */
X#ifndef ENUMS
X#define ONAME     'a'
X#define OREG 'b'
X#define REGVAR    'c'
X#define PUSHER    'd'
X#define CMPBR     'e'
X#define FIELDAS   'f'
X#else
Xenum {
X    ONAME = 'a', OREG, REGVAR, PUSHER, CMPBR, FIELDAS
X};
X#endif
X
X#define BR_TOK    'i'
X
X/* evaluation order values */
X#define EV_NONE       0
X#define EV_LEFT       1
X#define EV_RIGHT  2
X#define EV_LR         3
X#define EV_RL         4
X#define EV_LRSEP  5    /* left then right, but seperate */
END_OF_FILE
if test 1400 -ne `wc -c <'hcc/GEN.H'`; then
    echo shar: \"'hcc/GEN.H'\" unpacked with wrong size!
fi
# end of 'hcc/GEN.H'
fi
if test -f 'hcc/MAKEFILE' -a "${1}" != "-c" ; then
```

```
  echo shar: Will not clobber existing file \"'hcc/MAKEFILE'\"
else
echo shar: Extracting \"'hcc/MAKEFILE'\" \(359 characters\)
sed "s/^X//" >'hcc/MAKEFILE' <<'END_OF_FILE'
X#RM=rm
XCC=cc
XLD=ld
XAS=jas
X
XOBJS =  d2.o decl.o expr.o fix.o fun.o g2.o gen.o gsub.o gunk.o \
X    main.o md.o nodes.o out_st.o p2.o pre.o tok.o subs_tos.o
X
XH = bstok.h cookie.h flags.h gen.h nodes.h ops.h param.h \
X    sttok.h tok.h tytok.h
X
XCFLAGS = -v -O
X
Xnhcc.ttp: $(OBJS)
X    ld -vpb -f tos.ld -o nhcc.ttp
X
X$(OBJS) : $(H)
X
Xclean:
X    $(RM) *.o
X
Xclobber:
X    $(RM) *.o hcc
END_OF_FILE
if test 359 -ne `wc -c <'hcc/MAKEFILE'`; then
    echo shar: \"'hcc/MAKEFILE'\" unpacked with wrong size!
fi
# end of 'hcc/MAKEFILE'
fi
if test -f 'hcc/OPS.H' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/OPS.H'\"
else
echo shar: Extracting \"'hcc/OPS.H'\" \(1058 characters\)
sed "s/^X//" >'hcc/OPS.H' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  ops.h
X *
X *  defines of allowed operands
X *  'E' means Dreg or OREG or ONAME or (if on left) ICON
X *  'Q' means ICON which is 1 to 8
```

```
X *  ONE means ICON which is 1
X */
X
X#define DOPD 1
X#define AOPD 2
X#define MOPD 4
X#define IOPD 8
X
X#define DOPA 0x10
X#define AOPA 0x20
X#define MOPA 0x40
X#define IOPA 0x80
X
X#define DOPM 0x100
X#define AOPM 0x200
X#define MOPM 0x400
X#define IOPM 0x800
X
X#define QOPD 0x1000
X#define ONEOPM    0x2000
X#define ASSOC     0x4000
X
X#define EOPD (DOPD|MOPD|IOPD)
X#define DOPE (DOPD|DOPM)
X#define IOPE (IOPD|IOPM)
X#define EOPA (DOPA|MOPA|IOPA)
END_OF_FILE
if test 1058 -ne `wc -c <'hcc/OPS.H'`; then
    echo shar: \"'hcc/OPS.H'\" unpacked with wrong size!
fi
# end of 'hcc/OPS.H'
fi
if test -f 'hcc/PARAM.H' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/PARAM.H'\"
else
echo shar: Extracting \"'hcc/PARAM.H'\" \(2330 characters\)
sed "s/^X//" >'hcc/PARAM.H' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  param.h
X *
X *  basic parameters
X */
X
X#ifndef MMCC
```

```
X#define MMCC 0
X#endif
X
X#ifndef CC68
X#define CC68 0
X#endif
X
X#ifndef dLibs
X#define dLibs     0
X#endif
X
X#undef NULL
X#define NULL 0L
X
X#ifndef LAT_HOST
X#ifndef ACK_HOST
X#define ENUMS     1    /* define if host compiler can do enums */
X#endif
X#endif
X
X#define DEBUG     1
X
X#define NHASH     32
X
X#define ROPEN     "r"
X#define WOPEN     "w"
X#define FLTFORM "%.3g"
X
X#define MAXSTR    256
X#define MAXINCL 10
X#define NMSIZE    12   /* must be at least 12 */
X#define NMXSIZE   32
X
X/* sizes of basic types */
X#ifndef NOLONGS
X#define SIZE_C    1    /* char */
X#define SIZE_I    2    /* int */
X#define SIZE_S    2    /* short */
X#define SIZE_L    4    /* long */
X#define SIZE_U    2    /* unsigned */
X#define SIZE_F    4    /* float */
X#define SIZE_D    4    /* double */
X#define SIZE_P    4    /* pointer */
X#define MAXINT    0x7fffL
X#define MININT    -0x8000L
X#define MAXUNS    0xffffL
X#else
X#define SIZE_C    1    /* char */
X#define SIZE_I    4    /* int */
X#define SIZE_S    2    /* short */
X#define SIZE_L    4    /* long */
X#define SIZE_U    4    /* unsigned */
X#define SIZE_F    4    /* float */
```

```
X#define SIZE_D    4    /* double */
X#define SIZE_P    4    /* pointer */
X#define MAXINT    0x7fffffff
X#endif
X
X/* alignment requirements */
X#define ALN_C     0    /* char */
X#define ALN_I     1    /* int */
X#define ALN_S     1    /* short */
X#define ALN_L     1    /* long */
X#define ALN_U     1    /* unsigned */
X#define ALN_F     1    /* float */
X#define ALN_D     1    /* double */
X#define ALN_P     1    /* pointer */
X
X#define STACKALN 1
X
X/* usage for registers */
X#ifndef OUT_AZ
X#define DRV_START 3
X#define DRV_END       7
X#define AREG      8
X#define ARV_START AREG+3
X#define ARV_END       AREG+5
X#define FRAMEP        AREG+6
X#else
X#define DRV_START 4
X#define DRV_END       7
X#define AREG      8
X#define ARV_START AREG+3
X#define ARV_END       AREG+4
X#define FRAMEP        AREG+5
X#endif
X
X/* definition of stack frame */
X#define ARG_BASE  8
X#define LOC_BASE  0
X
X/* need for own buffers (ALCYON hack ) ? */
X#define NEEDBUF       0
X
END_OF_FILE
if test 2330 -ne `wc -c <'hcc/PARAM.H'`; then
    echo shar: \"'hcc/PARAM.H'\" unpacked with wrong size!
fi
# end of 'hcc/PARAM.H'
fi
if test -f 'hcc/STTOK.H' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/STTOK.H'\"
else
echo shar: Extracting \"'hcc/STTOK.H'\" \(1009 characters\)
sed "s/^X//" >'hcc/STTOK.H' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
```

```
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  sttok.h
X *
X *  statement keyword tokens
X */
X
X#ifndef ENUMS
X#define K_GOTO        'a'
X#define K_RETURN  'b'
X#define K_BREAK       'c'
X#define K_CONTINUE    'd'
X
X#define K_IF      'e'
X#define K_ELSE        'f'
X#define K_FOR         'g'
X#define K_DO      'h'
X#define K_WHILE       'i'
X#define K_SWITCH  'j'
X
X#define K_CASE        'k'
X#define K_DEFAULT 'l'
X#define K_ASM         'm'
X
X#define K_SIZEOF  'n'
X
X#else
Xenum {
X    K_GOTO = 'a', K_RETURN, K_BREAK, K_CONTINUE,
X    K_IF, K_ELSE, K_FOR, K_DO, K_WHILE, K_SWITCH,
X    K_CASE, K_DEFAULT, K_ASM,
X    K_SIZEOF,
X};
X#endif
END_OF_FILE
if test 1009 -ne `wc -c <'hcc/STTOK.H'`; then
    echo shar: \"'hcc/STTOK.H'\" unpacked with wrong size!
fi
# end of 'hcc/STTOK.H'
fi
if test -f 'hcc/SUBS_TOS.S' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/SUBS_TOS.S'\"
else
echo shar: Extracting \"'hcc/SUBS_TOS.S'\" \(492 characters\)
sed "s/^X//" >'hcc/SUBS_TOS.S' <<'END_OF_FILE'
X    .globl _lclr
```

```
X    .text
X_lclr:
X    link a6,#0
X
X    move.l   8(a6),a0
X    move 12(a6),d0
X    bra  L5
XL4:
X    clr.l    (a0)+
XL5:
X    dbf  d0,L4
X
X    unlk a6
X    rts
X
X    .globl _lcpy
X    .text
X_lcpy:
X    link a6,#0
X
X    move.l   8(a6),a0
X    move.l   12(a6),a1
X    move 16(a6),d0
X    bra  L8
XL7:
X    move.l   (a1)+,(a0)+
XL8:
X    dbf  d0,L7
X
X    unlk a6
X    rts
X
X    .data
X    .globl   _f_zero
X_f_zero:
X    .dc.l    0
X    .globl   _f_ten
X_f_ten:
X    .dc.l    $a0000044
X    .globl   _f_tenth
X_f_tenth:
X    .dc.l    $cccccd3d
X    .globl   _f_e10
X_f_e10:
X    .dc.l    $9502f962
X    .globl   _f_em10
X_f_em10:
X    .dc.l    $dbe6fd1f
END_OF_FILE
if test 492 -ne `wc -c <'hcc/SUBS_TOS.S'`; then
    echo shar: \"'hcc/SUBS_TOS.S'\" unpacked with wrong size!
fi
# end of 'hcc/SUBS_TOS.S'
```

```
fi
if test -f 'hcc/TOK.H' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/TOK.H'\"
else
echo shar: Extracting \"'hcc/TOK.H'\" \(1211 characters\)
sed "s/^X//" >'hcc/TOK.H' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X * 
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  tok.h
X *
X *  establish token values
X */
X
Xstruct tok {
X    char *name;
X    int  tnum;
X    long ival;
X    double   fval;
X    char prec;
X    int  flags;
X};
X
X#include "flags.h"
X#include "bstok.h"
X
X#define is_sclass(x)  (x >= K_EXTERN && x <= K_STATIC)
X#define is_tadj(x)    (x >= K_LONG && x <= K_UNSIGNED)
X#define is_btype(x)   (x >= K_INT && x <= K_VOID)
X#define is_tykw(x)    (x >= K_EXTERN && x <= K_STRUCT)
X
X#define is_stkw(x)    (x >= K_GOTO && x <= K_ASM)
X#define is_brast(x)   (x >= K_GOTO && x <= K_CONTINUE)
X#define is_blkst(x)   (x >= K_IF && x <= K_SWITCH)
X#define is_lblst(x)   (x >= K_CASE && x <= K_DEFAULT)
X
X#include "tytok.h"
X#include "sttok.h"
X
X/* modifier seen flags */
X#define SAW_SHORT 1
X#define SAW_LONG  2
X#define SAW_UNS       4
END_OF_FILE
if test 1211 -ne `wc -c <'hcc/TOK.H'`; then
    echo shar: \"'hcc/TOK.H'\" unpacked with wrong size!
```

```
fi
# end of 'hcc/TOK.H'
fi
if test -f 'hcc/TOS.LD' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/TOS.LD'\"
else
echo shar: Extracting \"'hcc/TOS.LD'\" \(176 characters\)
sed "s/^X//" >'hcc/TOS.LD' <<'END_OF_FILE'
X\sozobon\lib\dstart.o
Xd2.o decl.o expr.o fix.o fun.o g2.o gen.o gsub.o gunk.o
Xmain.o md.o nodes.o out_st.o p2.o pre.o tok.o subs_tos.o
X\sozobon\lib\libm.a \sozobon\lib\dlibs.a
END_OF_FILE
if test 176 -ne `wc -c <'hcc/TOS.LD'`; then
    echo shar: \"'hcc/TOS.LD'\" unpacked with wrong size!
fi
# end of 'hcc/TOS.LD'
fi
if test -f 'hcc/TYTOK.H' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/TYTOK.H'\"
else
echo shar: Extracting \"'hcc/TYTOK.H'\" \(1490 characters\)
sed "s/^X//" >'hcc/TYTOK.H' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X * 
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  tytok.h
X *
X *  keyword token values
X */
X
X#ifndef ENUMS
X#define K_EXTERN  'A'
X#define K_AUTO        'B'
X#define K_REGISTER    'C'
X#define K_TYPEDEF 'D'
X#define K_STATIC  'E'
X
X#define ENUM_SC       K_STATIC+1    /* storage class for enum item */
X#define HERE_SC       K_STATIC+2    /* storage class for glb def */
X#define FIRST_SC  K_EXTERN
X#define LAST_SC       K_STATIC
X
X#define T_UCHAR       'F'
X#define T_ULONG       'G'
X
```

```
X#define K_LONG         'H'
X#define K_SHORT        'I'
X#define K_UNSIGNED     'J'
X
X#define K_INT          'K'
X#define K_CHAR         'L'
X#define K_FLOAT        'M'
X#define K_DOUBLE  'N'
X#define K_VOID         'O'
X
X#define FIRST_BAS T_UCHAR
X#define LAST_BAS  K_VOID
X
X#define K_UNION        'P'
X#define K_ENUM         'Q'
X#define K_STRUCT  'R'
X#else
X#define FIRST_SC  K_EXTERN
X#define LAST_SC        K_STATIC
X#define FIRST_BAS T_UCHAR
X#define LAST_BAS  K_VOID
Xenum {
X    K_EXTERN = 'A', K_AUTO, K_REGISTER, K_TYPEDEF, K_STATIC,
X    ENUM_SC, HERE_SC,
X    T_UCHAR, T_ULONG,
X    K_LONG, K_SHORT, K_UNSIGNED,
X    K_INT, K_CHAR, K_FLOAT, K_DOUBLE, K_VOID,
X    K_UNION, K_ENUM, K_STRUCT
X};
X#endif
END_OF_FILE
if test 1490 -ne `wc -c <'hcc/TYTOK.H'`; then
    echo shar: \"'hcc/TYTOK.H'\" unpacked with wrong size!
fi
# end of 'hcc/TYTOK.H'
fi
if test ! -d 'top' ; then
    echo shar: Creating directory \"'top'\"
    mkdir 'top'
fi
if test -f 'top/HEALTH.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/HEALTH.C'\"
else
echo shar: Extracting \"'top/HEALTH.C'\" \(3765 characters\)
sed "s/^X//" >'top/HEALTH.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
```

```
X *    of using this software, even if they result from defects in it.
X */
X
X/*
X * Routines for data flow analysis by block and function.
X */
X
X#include "top.h"
X
X/*
X * bprep() - initial analysis of a block
X */
Xvoid
Xbprep(bp)
XBLOCK   *bp;
X{
X    register INST *ip;
X    register int  ref, set;
X
X    ref = set = 0;
X    for (ip = bp->first; ip != NULL ;ip = ip->next) {
X         ref |= (ip->rref = reg_ref(ip)) & ~set;
X         set |= (ip->rset = reg_set(ip));
X    }
X    bp->rref = ref;
X    bp->rset = set;
X}
X
X/*
X * fprep() - perform initial analysis of individual blocks in a function
X */
Xstatic  void
Xfprep(bp)
XBLOCK   *bp;
X{
X    register BLOCK     *cp;
X
X    for (cp = bp; cp != NULL ;cp = cp->next)
X         bprep(cp);
X}
X
Xstatic  bool
Xscan_ref(bp, reg)
Xregister BLOCK   *bp;
Xregister int reg;
X{
X    register INST *ip;
X
X    if (bp == NULL)
X         return FALSE;
X
X    if (bp->rref & reg)
X         return TRUE;
```

```
X
X    if (bp->rset & reg)
X         return FALSE;
X
X    if (bp->flags & B_MARK)
X         return FALSE;
X
X    if (bp->flags & B_RET)
X         return FALSE;
X
X    bp->flags |= B_MARK;
X
X    if (scan_ref(bp->bcond, reg))
X         return TRUE;
X
X    if (scan_ref(bp->bfall, reg))
X         return TRUE;
X
X    for (ip = bp->first; ip != NULL ;ip = ip->next) {
X         if (ip->opcode == DC && (ip->flags & LENL)) {
X              BLOCK    *db;
X
X              if (is_astr(ip->src.amode) && (ip->src.astr != NULL) &&
X                  (db = getsym(ip->src.astr)) != NULL)
X                   if (scan_ref(db, reg))
X                        return TRUE;
X         }
X    }
X    return FALSE;
X}
X
X/*
X * is_live(bp, reg) - is 'reg' live at the end of block 'bp'
X *
X * Look ahead through the traversal graph to see what might happen to the
X * given register. If we can find any reachable block that references 'reg'
X * before setting it, the register is live. Scanning stops when the register
X * is set, we loop back to the starting block, or the function returns.
X */
Xstatic  bool
Xis_live(bp, reg)
Xregister BLOCK   *bp;
Xregister int reg;
X{
X    register INST *ip;
X    register BLOCK    *cp;
X
X    /*
X     * Clear all the mark bits
X     */
X    for (cp = fhead; cp != NULL ;cp = cp->next)
X         cp->flags &= ~B_MARK;
X
```

```
X    bp->flags |= B_MARK;
X
X    if (scan_ref(bp->bfall, reg))
X         return TRUE;
X
X    if (scan_ref(bp->bcond, reg))
X         return TRUE;
X
X    for (ip = bp->first; ip != NULL ;ip = ip->next) {
X         if (ip->opcode == DC && (ip->flags & LENL)) {
X              BLOCK    *db;
X
X              if (is_astr(ip->src.amode) && (ip->src.astr != NULL) &&
X                  (db = getsym(ip->src.astr)) != NULL)
X                   if (scan_ref(db, reg))
X                        return TRUE;
X         }
X    }
X    return FALSE;
X}
X
X/*
X * bflow() - live/dead analysis for a given block
X *
X * Work backward from the end of the block, checking the status of registers.
X * To start with, figure out the state of each register as of the end of the
X * block. Then work backward, checking registers only as necessary.
X */
Xstatic  void
Xbflow(bp)
XBLOCK   *bp;
X{
X    register INST *ip;
X    register int  live = 0;
X    register int  reg;
X
X    /*
X     * Figure out what's alive at the end of this block.
X     */
X    for (reg = FIRSTREG; reg <= LASTREG ;reg++) {
X         if (is_live(bp, RM(reg)))
X              live |= RM(reg);
X    }
X    /*
X     * Now work through the instructions in the block.
X     */
X    for (ip = bp->last; ip != NULL ;ip = ip->prev) {
X         ip->live = live;
X
X         for (reg = FIRSTREG; reg <= LASTREG ;reg++) {
X              if (ip->rref & RM(reg))
X                   live |= RM(reg);
X              else if (ip->rset & RM(reg))
```

```
X                   live &= ~RM(reg);
X         }
X    }
X}
X
Xvoid
Xrhealth(bp, do_prep)
XBLOCK    *bp;
Xbool     do_prep;
X{
X    register BLOCK     *cp;
X
X    if (do_prep)
X         fprep(bp);
X
X    for (cp = bp; cp != NULL ;cp = cp->next)
X         bflow(cp);
X}
END_OF_FILE
if test 3765 -ne `wc -c <'top/HEALTH.C'`; then
    echo shar: \"'top/HEALTH.C'\" unpacked with wrong size!
fi
# end of 'top/HEALTH.C'
fi
if test -f 'top/INST.H' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/INST.H'\"
else
echo shar: Extracting \"'top/INST.H'\" \(2922 characters\)
sed "s/^X//" >'top/INST.H' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X
X/*
X * Misc. define's, etc. for instruction parsing.
X */
X
Xstruct  opnd {
X    unsigned char amode;    /* addressing mode used */
X    unsigned char areg;     /* primary register */
X    unsigned char ireg;     /* index register, if applicable */
X    union {
X         long i_disp;       /* also used for immediate data */
X         char *i_astr;  /* pointer to any symbol present */
X    } i_data;
X};
```

```
X#define  disp i_data.i_disp
X#define  astr i_data.i_astr
X
X/*
X * is_astr(m)     macro to determine, based on the amode field, whether
X *       the 'astr' field of the structure is active.
X */
X#define  is_astr(m)    (((m) == ABS) || ((m) == (IMM|SYMB)))
X
X/*
X * Addressing modes (in 'amode')
X */
X#define  NONE 0         /* operand unused */
X#define  REG  1         /* register direct */
X#define  IMM  2         /* immediate */
X#define  ABS  3         /* absolute */
X#define  REGI 4         /* reg. indirect */
X#define  REGID    5         /* reg. indirect, w/ displacement */
X#define  REGIDX   6         /* reg. indirect, w/ displacement & index */
X#define  PCD  7         /* PC relative, w/ displacement */
X#define  PCDX 8         /* PC relative, w/ displacement & index */
X
X#define  XLONG    0x10      /* long index register used */
X#define  SYMB 0x20      /* symbol used, not constant */
X#define  INC  0x40      /* auto-increment */
X#define  DEC  0x80      /* auto-decrement */
X
X#define  MMASK    0x0f      /* mode mask */
X#define  FMASK    0xf0      /* flag mask */
X
X#define  M(x) ((x) & MMASK)
X#define  F(x) ((x) & FMASK)
X
X/*
X * Registers
X */
X
X#define  FIRSTREG 0
X#define  A0   0
X#define  A1   1
X#define  A2   2
X#define  A3   3
X#define  A4   4
X#define  A5   5
X#define  A6   6
X#define  A7   7
X#define  SP   7    /* alias for A7 */
X#define  D0   8
X#define  D1   9
X#define  D2   10
X#define  D3   11
X#define  D4   12
X#define  D5   13
```

```
X#define  D6   14
X#define  D7   15
X#define  LASTREG  15
X
X#define  ISD(x)    ((x) >= D0 && (x) <= D7)    /* is 'x' a  D reg. */
X#define  ISA(x)    ((x) >= A0 && (x) <= A7)    /* is 'x' an A reg. */
X#define  RM(x)     (1 << (x))             /* form a register mask */
X
X/*
X * DOK(x) - evaluates TRUE if 'x' is okay for a displacement value.
X *
X * 'x' must be of type "long"
X */
X#define  DOK(x)    (((x) >= -32768L) && ((x) <= 32767L))
X
X/*
X * D8OK(x) - like DOK but for 8-bit displacements
X */
X#define  D8OK(x)   (((x) >= -128) && ((x) <= 127))
X
Xstruct  inst {
X    char opcode;            /* type of instruction */
X    char flags;             /* length, etc. */
X    struct   opnd src, dst; /* optional operands */
X    int  rref, rset;
X    int  live;              /* regs. live after this inst. */
X    struct   inst *next;
X    struct   inst *prev;
X};
X
X/*
X * Instruction flags
X */
X
X#define  LENB 0x01
X#define  LENW 0x02
X#define  LENL 0x04
END_OF_FILE
if test 2922 -ne `wc -c <'top/INST.H'`; then
    echo shar: \"'top/INST.H'\" unpacked with wrong size!
fi
# end of 'top/INST.H'
fi
if test -f 'top/IO.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/IO.C'\"
else
echo shar: Extracting \"'top/IO.C'\" \(2661 characters\)
sed "s/^X//" >'top/IO.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
```

```
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X#include "top.h"
X
X/*
X * Low-level i/o routines.
X */
X
X/*
X * mode tells what kind of stuff we're reading at the moment.
X */
Xstatic  int  mode;
X
X#define BSS  0
X#define DATA 1
X#define TEXT 2
X
Xstatic  char *mnames[] = {
X    ".bss",
X    ".data",
X    ".text"
X};
X
X/*
X * Tokens from the current line...
X */
Xchar    *t_line;      /* the entire line */
Xchar    *t_lab;            /* label, if any */
Xchar    *t_op;             /* opcode */
Xchar    *t_arg;            /* arguments */
X
X#define ISWHITE(c)    ((c) == '\t' || (c) == ' ' || (c) == '\n')
X
X#define LSIZE    2048 /* max. size of an input line */
X
X/*
X * readline() - read the next line from the file
X *
X * readline passes data and bss through to the output, only returning
X * when a line of text has been read. Returns FALSE on end of file.
X */
Xbool
Xreadline()
X{
X    char *fgets();
X    static    void tokenize();
X    static    char buf[LSIZE];
X
X    /*
X     * Keep looping until we get a line of text
```

```
X     */
X    for (;;) {
X         if (fgets(buf, LSIZE, ifp) == NULL)
X              return FALSE;
X
X         t_line = buf;
X
X         /*
X          * Find out if the mode is changing.
X          */
X         tokenize(buf);
X
X         /*
X          * If we see a "var" hint from the compiler, call addvar()
X          * to remember it for later use.
X          */
X         if (t_lab[0] == ';') {
X              if (strcmp(t_lab, ";var") == 0)
X                   addvar(atoi(t_op), atoi(t_arg));
X              continue;
X         }
X
X         if (t_op[0] == '.') {  /* is it a pseudo-op? */
X              if (strcmp(t_op, mnames[BSS]) == 0)
X                   mode = BSS;
X              else if (strcmp(t_op, mnames[DATA]) == 0)
X                   mode = DATA;
X              else if (strcmp(t_op, mnames[TEXT]) == 0) {
X                   mode = TEXT;
X                   continue;
X              }
X         }
X         if (mode == TEXT)
X              return TRUE;
X         else
X              fputs(buf, ofp);
X    }
X}
X
Xstatic void
Xtokenize(s)
Xregister char     *s;
X{
X    static    char label[LSIZE], opcode[LSIZE], args[LSIZE];
X    register int  i;
X
X    /*
X     * Grab the label, if any
X     */
X    i = 0;
X    while (*s && !ISWHITE(*s) && *s != ':')
X         label[i++] = *s++;
X    label[i] = '\0';
```

```
X
X    if (*s == ':')
X         s++;
X
X    while (ISWHITE(*s))
X         s++;
X
X    /*
X     * Grab the opcode
X     */
X    i = 0;
X    while (*s && !ISWHITE(*s))
X         opcode[i++] = *s++;
X    opcode[i] = '\0';
X
X    while (ISWHITE(*s))
X         s++;
X
X    /*
X     * Grab the arguments
X     */
X    i = 0;
X    while (*s && !ISWHITE(*s))
X         args[i++] = *s++;
X    args[i] = '\0';
X
X    t_lab = label;
X    t_op = opcode;
X    t_arg = args;
X}
END_OF_FILE
if test 2661 -ne `wc -c <'top/IO.C'`; then
    echo shar: \"'top/IO.C'\" unpacked with wrong size!
fi
# end of 'top/IO.C'
fi
if test -f 'top/MAIN.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/MAIN.C'\"
else
echo shar: Extracting \"'top/MAIN.C'\" \(3815 characters\)
sed "s/^X//" >'top/MAIN.C' <<'END_OF_FILE'
X/* Copyright (c) 1988,1989 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X#include "top.h"
X
```

```
XFILE    *ifp, *ofp;        /* input/output file pointers */
X
X#ifndef MINIX
Xlong    _STKSIZ = 32768L;  /* need mucho stack for recursion */
X#endif
X
X/*
X * Options
X */
Xbool    debug   = FALSE;
Xbool    do_brev = TRUE;         /* branch reversals enabled */
Xbool    do_peep = TRUE;         /* peephole optimizations enabled */
Xbool    do_regs = TRUE;         /* do "registerizing" */
Xbool    do_lrot = TRUE;         /* do loop rotations */
Xbool    verbose = FALSE;
X
X/*
X * Optimization statistics (use -v to print)
X */
Xint s_bdel = 0;         /* branches deleted */
Xint s_badd = 0;         /* branches added */
Xint s_brev = 0;         /* branch reversals */
Xint s_lrot = 0;         /* loop rotations */
Xint s_peep1 = 0;        /* 1 instruction peephole changes */
Xint s_peep2 = 0;        /* 2 instruction peephole changes */
Xint s_peep3 = 0;        /* 3 instruction peephole changes */
Xint s_idel = 0;         /* instructions deleted */
Xint s_reg = 0;          /* variables "registerized" */
X
X#define  TMPFILE   "top_tmp.$$$" /* temporary file name */
Xint use_temp = FALSE;  /* using temporary file */
X
Xchar    *Version =
X"top Version 1.20  Copyright (c) 1988,1989 by Sozobon, Limited.";
X
Xusage()
X{
X    fprintf(stderr, "usage: top [-dvblpr] infile [outfile]\n");
X    exit(1);
X}
X
Xmain(argc, argv)
Xint argc;
Xchar    *argv[];
X{
X    FILE *fopen();
X    register char *s;
X
X    while (argc > 1 && argv[1][0] == '-') {
X         for (s = &argv[1][1]; *s ;s++) {
X              switch (*s) {
X              case 'd':
X                   debug = TRUE;
```

```
X                 break;
X            case 'b':
X                 do_brev = FALSE;
X                 break;
X            case 'p':
X                 do_peep = FALSE;
X                 break;
X            case 'r':
X                 do_regs = FALSE;
X                 break;
X            case 'l':
X                 do_lrot = FALSE;
X                 break;
X            case 'v':
X                 fprintf(stderr, "%s\n", Version);
X                 verbose = TRUE;
X                 break;
X            case 'O':
X                 /*
X                  * When options are received from 'cc' they
X                  * look like "-Oxxx", so just ignore the 'O'.
X                  */
X                 break;
X            default:
X                 usage();
X                 break;
X            }
X        }
X        argv++;
X        argc--;
X    }
X
X    if (argc > 3)
X        usage();
X
X    if (argc > 1) {
X        if (strcmp(argv[1], "-") == 0)
X            ifp = stdin;
X        else if ((ifp = fopen(argv[1], "r")) == NULL) {
X            fprintf(stderr, "top: can't open input file '%s'\n",
X                 argv[1]);
X            exit(1);
X        }
X        if (argc > 2) {
X            if (strcmp(argv[2], "-") == 0)
X                 ofp = stdout;
X            else if ((ofp = fopen(argv[2], "w")) == NULL) {
X                 fprintf(stderr, "top: can't open output file '%s'\n",
X                      argv[2]);
X                 exit(1);
X            }
X        } else {
X            if ((ofp = fopen(TMPFILE, "w")) == NULL) {
```

```
X                 fprintf(stderr, "top: can't create temp file\n");
X                 exit(1);
X             }
X             use_temp = TRUE;
X         }
X    } else
X         usage();
X
X    dofile();
X
X    if (verbose) {
X         if (do_peep) {
X             fprintf(stderr, "Peephole changes (1): %4d\n", s_peep1);
X             fprintf(stderr, "Peephole changes (2): %4d\n", s_peep2);
X             fprintf(stderr, "Peephole changes (3): %4d\n", s_peep3);
X             fprintf(stderr, "Instructions deleted: %4d\n", s_idel);
X         }
X         if (do_regs)
X             fprintf(stderr, "Variables registered: %4d\n", s_reg);
X         if (do_lrot)
X             fprintf(stderr, "Loop rotations      : %4d\n", s_lrot);
X         if (do_brev)
X             fprintf(stderr, "Branch reversals    : %4d\n", s_brev);
X         fprintf(stderr, "Branches removed    : %4d\n", s_bdel - s_badd);
X    }
X
X    /*
X     * If we're overwriting the original file, remove the old
X     * version, and rename the temp file to the old name.
X     */
X    if (use_temp) {
X         remove(argv[1]);
X         rename(TMPFILE, argv[1]);
X    }
X
X    exit(0);
X}
X
Xdofile()
X{
X    if (!readline())
X         return;
X
X    while (dofunc())
X         ;
X}
END_OF_FILE
if test 3815 -ne `wc -c <'top/MAIN.C'`; then
    echo shar: \"'top/MAIN.C'\" unpacked with wrong size!
fi
# end of 'top/MAIN.C'
fi
if test -f 'top/MAKEFILE' -a "${1}" != "-c" ; then
```

```
  echo shar: Will not clobber existing file \"'top/MAKEFILE'\"
else
echo shar: Extracting \"'top/MAKEFILE'\" \(260 characters\)
sed "s/^X//" >'top/MAKEFILE' <<'END_OF_FILE'
X#
X# Makefile for the optimizer
X#
XCC=cc
X
XOBJ =   main.o io.o util.o sym.o func.o branch.o inst.o opcodes.o \
X   health.o data.o peep1.o peep2.o peep3.o reg.o
X
XCFLAGS = -O
X
Xxtop.ttp : $(OBJ)
X   $(CC) $(OBJ) -o xtop.ttp
X
Xclean:
X   $(RM) *.o
X
Xclobber:
X   $(RM) *.o xtop.ttp
END_OF_FILE
if test 260 -ne `wc -c <'top/MAKEFILE'`; then
    echo shar: \"'top/MAKEFILE'\" unpacked with wrong size!
fi
# end of 'top/MAKEFILE'
fi
if test -f 'top/OPCODES.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/OPCODES.C'\"
else
echo shar: Extracting \"'top/OPCODES.C'\" \(1545 characters\)
sed "s/^X//" >'top/OPCODES.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X
X/*
X * Mnemonics for the instructions.
X */
X
Xchar    *opnames[] = {
X   "or",
X   "btst",
X   "movep",
X   "bchg",
```

```
X    "bclr",
X    "bset",
X    "and",
X    "sub",
X    "add",
X    "eor",
X    "cmp",
X    "move",
X    "negx",
X    "chk",
X    "lea",
X    "clr",
X    "neg",
X    "not",
X    "nbcd",
X    "swap",
X    "pea",
X    "ext",
X    "movem",
X    "tst",
X    "tas",
X    "trap",
X    "link",
X    "unlk",
X    "reset",
X    "nop",
X    "stop",
X    "rte",
X    "rts",
X    "trapv",
X    "rtr",
X    "jsr",
X    "jmp",
X    "addq",
X    "st",
X    "dbt",
X    "subq",
X    "sf",
X    "dbra",       /* also dbf */
X    "shi",
X    "dbhi",
X    "sls",
X    "dbls",
X    "scc",
X    "dbcc",
X    "scs",
X    "dbcs",
X    "sne",
X    "dbne",
X    "seq",
X    "dbeq",
X    "svc",
X    "dvc",
```

```
X    "svs",
X    "dvs",
X    "spl",
X    "smi",
X    "dbmi",
X    "sge",
X    "dbge",
X    "slt",
X    "dblt",
X    "sgt",
X    "dbgt",
X    "sle",
X    "dble",
X    "bra",
X    "bsr",
X    "bhi",
X    "bls",
X    "bcc",
X    "bcs",
X    "bne",
X    "beq",
X    "bvc",
X    "bvs",
X    "bpl",
X    "bmi",
X    "bge",
X    "blt",
X    "bgt",
X    "ble",
X    "moveq",
X    "divu",
X    "sbcd",
X    "divs",
X    "subx",
X    "cmpm",
X    "mulu",
X    "abcd",
X    "exg",
X    "muls",
X    "addx",
X    "asr",
X    "lsr",
X    "roxr",
X    "ror",
X    "asl",
X    "lsl",
X    "roxl",
X    "rol",
X/*
X * Pseudo-ops that show up in text sections
X */
X    ".dc",
X    (char *) 0
```

```
X};
END_OF_FILE
if test 1545 -ne `wc -c <'top/OPCODES.C'`; then
    echo shar: \"'top/OPCODES.C'\" unpacked with wrong size!
fi
# end of 'top/OPCODES.C'
fi
if test -f 'top/OPCODES.H' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/OPCODES.H'\"
else
echo shar: Extracting \"'top/OPCODES.H'\" \(2212 characters\)
sed "s/^X//" >'top/OPCODES.H' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X
X/*
X * Defines for the instructions
X */
X
X#define OR   0
X#define BTST 1
X#define MOVEP    2
X#define BCHG 3
X#define BCLR 4
X#define BSET 5
X#define AND  6
X#define SUB  7
X#define ADD  8
X#define EOR  9
X#define CMP  10
X#define MOVE 11
X#define NEGX 12
X#define CHK  13
X#define LEA  14
X#define CLR  15
X#define NEG  16
X#define NOT  17
X#define NBCD 18
X#define SWAP 19
X#define PEA  20
X#define EXT  21
X#define MOVEM    22
X#define TST  23
X#define TAS  24
X#define TRAP 25
```

```
X#define LINK 26
X#define UNLK 27
X#define RESET    28
X#define NOP  29
X#define STOP 30
X#define RTE  31
X#define RTS  32
X#define TRAPV    33
X#define RTR  34
X#define JSR  35
X#define JMP  36
X#define ADDQ 37
X#define ST   38
X#define DBT  39
X#define SUBQ 40
X#define SF   41
X#define DBRA 42   /* aka DBF */
X#define SHI  43
X#define DBHI 44
X#define SLS  45
X#define DBLS 46
X#define SCC  47
X#define DBCC 48
X#define SCS  49
X#define DBCS 50
X#define SNE  51
X#define DBNE 52
X#define SEQ  53
X#define DBEQ 54
X#define SVC  55
X#define DVC  56
X#define SVS  57
X#define DVS  58
X#define SPL  59
X#define SMI  60
X#define DBMI 61
X#define SGE  62
X#define DBGE 63
X#define SLT  64
X#define DBLT 65
X#define SGT  66
X#define DBGT 67
X#define SLE  68
X#define DBLE 69
X#define BRA  70
X#define BSR  71
X#define BHI  72
X#define BLS  73
X#define BCC  74
X#define BCS  75
X#define BNE  76
X#define BEQ  77
X#define BVC  78
```

```
X#define  BVS  79
X#define  BPL  80
X#define  BMI  81
X#define  BGE  82
X#define  BLT  83
X#define  BGT  84
X#define  BLE  85
X#define  MOVEQ    86
X#define  DIVU 87
X#define  SBCD 88
X#define  DIVS 89
X#define  SUBX 90
X#define  CMPM 91
X#define  MULU 92
X#define  ABCD 93
X#define  EXG  94
X#define  MULS 95
X#define  ADDX 96
X#define  ASR  97
X#define  LSR  98
X#define  ROXR 99
X#define  ROR  100
X#define  ASL  101
X#define  LSL  102
X#define  ROXL 103
X#define  ROL  104
X/*
X * Pseudo-ops
X */
X#define  DC   105
END_OF_FILE
if test 2212 -ne `wc -c <'top/OPCODES.H'`; then
    echo shar: \"'top/OPCODES.H'\" unpacked with wrong size!
fi
# end of 'top/OPCODES.H'
fi
if test -f 'top/SYM.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/SYM.C'\"
else
echo shar: Extracting \"'top/SYM.C'\" \(3184 characters\)
sed "s/^X//" >'top/SYM.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X#include "top.h"
X
```

```
X/*
X * Symbol table:
X *
X * For each symbol, contains a pointer to the block starting at the
X * given symbol, and a pointer to the next symbol in the symbol table.
X */
Xstruct  sym {
X    char *name;
X    BLOCK    *bp;
X    struct   sym  *next;
X};
X
Xstruct  sym  *sfirst = NULL;
Xstruct  sym  *slast  = NULL;
X
X/*
X * newblock(name) - allocate a new block structure and initialize it
X */
XBLOCK *
Xnewblock(name)
Xregister char    *name;
X{
X    register BLOCK    *bp;
X
X    if ((bp = (BLOCK *) alloc(sizeof(BLOCK))) == NULL)
X        return NULL;
X
X    /*
X     * Initialize the allocated block structure.
X     */
X
X    if ((bp->name = strsave(name)) == NULL) {
X        free(bp);
X        return NULL;
X    }
X
X    bp->flags = 0;
X    bp->ref   = 0;
X    bp->bcode = 0;
X    bp->first = NULL;
X    bp->last  = NULL;
X    bp->bcond = NULL;
X    bp->bfall = NULL;
X    bp->chain = NULL;
X    bp->next  = NULL;
X    bp->rref = bp->rset = 0;
X
X    return bp;
X}
X
X/*
X * mksym(name) - make a new symbol table entry
X *
```

```
X * mksym creates new symbol table entries, and allocates space for the
X * 'block' structure that will be used for the symbol. This can happen
X * when a reference to a block is detected, but before the block has
X * been encountered. Since we allocate the block structure here, other
X * blocks can reference it before we actually read it in.
X */
XBLOCK *
Xmksym(name)
Xregister char     *name;
X{
X    register struct    sym  *new;
X
X    if ((new = (struct sym *) alloc(sizeof(struct sym))) == NULL)
X         return NULL;
X
X    if ((new->bp = newblock(name)) == NULL) {
X         free(new);
X         return NULL;
X    }
X
X    new->name = new->bp->name;
X    new->next = NULL;
X
X    if (sfirst == NULL)
X         sfirst = slast = new;
X    else {
X         slast->next = new;
X         slast = new;
X    }
X
X    return new->bp;
X}
X
X/*
X * getsym(name) - return a pointer to the block for symbol 'name'
X *
X * Scans the symbol table for the given symbol and returns a pointer
X * to its block, when found, or NULL if not present.
X */
XBLOCK *
Xgetsym(name)
Xregister char     *name;
X{
X    register struct    sym  *sp;
X
X    for (sp = sfirst; sp != NULL ;sp = sp->next) {
X         if (strcmp(sp->name, name) == 0)
X              return sp->bp;
X    }
X    return NULL;
X}
X
X/*
```

```
X * freeop() - free an operand
X */
Xvoid
Xfreeop(op)
Xstruct  opnd *op;
X{
X    if (is_astr(op->amode) && op->astr != NULL)
X         free(op->astr);
X}
X
X/*
X * freesym() - free all symbol table space
X */
Xvoid
Xfreesym()
X{
X    register struct    sym  *sp, *nexts;
X    register INST *ip, *nexti;
X
X    for (sp = sfirst; sp != NULL ;sp = nexts) {
X         nexts = sp->next;
X         for (ip = sp->bp->first; ip != NULL ; ip = nexti) {
X              nexti = ip->next;
X
X              freeop(&ip->src);
X              freeop(&ip->dst);
X              free(ip);
X         }
X         free(sp->name);
X         free(sp->bp);
X         free(sp);
X    }
X    sfirst = slast = NULL;
X}
X
Xchar *
Xmktmp()
X{
X    static    char tname[32];
X    static    int  tnum = 0;
X
X    sprintf(tname, "T%d", tnum++);
X
X    return tname;
X}
END_OF_FILE
if test 3184 -ne `wc -c <'top/SYM.C'`; then
    echo shar: \"'top/SYM.C'\" unpacked with wrong size!
fi
# end of 'top/SYM.C'
fi
if test -f 'top/TOP.H' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/TOP.H'\"
```

```
else
echo shar: Extracting \"'top/TOP.H'\" \(3752 characters\)
sed "s/^X//" >'top/TOP.H' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X
X#include <stdio.h>
X#include <ctype.h>
X
X#include "inst.h"
X#include "opcodes.h"
X
X/* #define   DEBUG         /* enable debug code */
X
X#ifdef  DEBUG
X#define DBG(x)        if (debug) { x; }
X#else
X#define DBG(x)
X#endif
X
X#ifndef void
X#define void int
X#endif
X
X/*
X * Basic defines and declarations for the optimizer.
X */
X
Xtypedef int  bool;
X
X#ifndef FALSE
X#define FALSE    0
X#define TRUE 1
X#endif
X
X/*
X * Basic Block:
X *
X * References a linked list of instructions that make up the block.
X * Each block can be exited via one of two branches, which are
X * represented by pointers to two other blocks, or null.
X */
Xstruct  block {
X    int  flags;            /* flags relating to this block */
X    int  ref;          /* # of references to this block */
```

```
X    int  bcode;            /* type of exiting branch */
X    char *name;            /* symbol name that starts the block */
X
X    struct   inst *first,       /* first instruction in block */
X             *last;        /* last instruction in block */
X
X    /*
X     * Execution traversals
X     */
X    struct   block    *bcond,       /* conditional branch (or NULL) */
X             *bfall;       /* "fall through" branch */
X
X    /*
X     * Logical traversals
X     */
X    struct   block    *chain;       /* links all blocks together */
X    struct   block    *next;        /* next block in the file */
X
X    /*
X     * Information for data-flow analysis
X     */
X    int  rref;             /* registers ref'd before set */
X    int  rset;             /* registers modified by block */
X};
X
Xtypedef struct block  BLOCK;
Xtypedef struct inst   INST;
X
X/*
X * Block flags
X */
X
X#define  B_GLOBAL  0x01     /* is the block's symbol global? */
X#define  B_TOUCHED 0x02     /* used in traversals */
X#define  B_LABEL       0x04     /* the block needs a label */
X#define  B_ISREACHED   0x08     /* block IS reached (for switches) */
X#define  B_RET         0x10     /* block terminates with a 'return' */
X#define  B_MARK        0x20     /* temporary 'touched' mark */
X
X/*
X * Global data
X */
X
Xextern   FILE *ifp, *ofp;       /* input and output file pointers */
X
X/*
X * Option flags set in main
X */
Xextern   bool debug;
Xextern   bool do_peep;      /* enable peephole opt. */
Xextern   bool do_brev;      /* enable branch reversals */
Xextern   bool do_regs;      /* enable "registerizing" */
Xextern   bool do_lrot;      /* enable loop rotations */
```

```
Xextern  bool verbose;
X
X/*
X * Optimization stats
X */
Xextern  int  s_bdel;
Xextern  int  s_badd;
Xextern  int  s_brev;
Xextern  int  s_peep1;
Xextern  int  s_peep2;
Xextern  int  s_peep3;
Xextern  int  s_idel;
Xextern  int  s_reg;
Xextern  int  s_lrot;
X
X/*
X * These are set after calling readline.
X */
Xextern  char *t_line;  /* text of the last line */
Xextern  char *t_lab;       /* label (if any) on the last line */
Xextern  char *t_op;        /* opcode */
Xextern  char *t_arg;       /* arguments */
X
X
Xextern  char *opnames[];   /* mnemonics for the instructions */
X
Xextern  BLOCK    *fhead;       /* head of the current function */
X
X/*
X * Function declarations
X */
X
X/*
X * branch.c
X */
Xextern  void bopt();
X
X/*
X * data.c
X */
Xextern  int  reg_ref(), reg_set();
Xextern  bool sets(), refs(), uses();
X
X/*
X * health.c
X */
Xextern  void rhealth(), bprep();
X
X/*
X * inst.c
X */
Xextern  void addinst(), delinst(), putinst();
Xextern  bool opeq();
```

```
X
X/*
X * io.c
X */
Xextern   bool readline();
X
X/*
X * peep1.c
X */
Xextern   void peep();
X
X/*
X * reg.c
X */
Xextern   void addvar(), setreg(), clrvar();
X
X/*
X * util.c
X */
Xextern   char *alloc();
Xextern   char *strsave();
X
X/*
X * sym.c
X */
Xextern   void freeop(), freesym();
Xextern   BLOCK    *getsym(), *mksym();
Xextern   char *mktmp();
END_OF_FILE
if test 3752 -ne `wc -c <'top/TOP.H'`; then
    echo shar: \"'top/TOP.H'\" unpacked with wrong size!
fi
# end of 'top/TOP.H'
fi
if test -f 'top/UTIL.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/UTIL.C'\"
else
echo shar: Extracting \"'top/UTIL.C'\" \(1057 characters\)
sed "s/^X//" >'top/UTIL.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X
X#include <stdio.h>
X
X/*
```

```
X * strsave(s) - copy s to dynamically allocated space
X */
Xchar *
Xstrsave(s)
Xregister char     *s;
X{
X    char *malloc(), *strcpy();
X
X    return strcpy(malloc((unsigned) (strlen(s) + 1)), s);
X}
X
X/*
X * alloc() - malloc with error checking
X */
Xchar *
Xalloc(n)
Xint n;
X{
X    extern    char *malloc();
X    char *p;
X
X    if ((p = malloc(n)) == NULL) {
X         fprintf(stderr, "top: out of memory\n");
X         exit(1);
X    }
X    return p;
X}
X
X#ifdef  MINIX
X
Xremove(f)
Xchar    *f;
X{
X    unlink(f);
X}
X
Xrename(f1, f2)
Xchar    *f1, *f2;
X{
X    unlink(f2);
X    link(f1, f2);
X    unlink(f1);
X}
X
X#endif
END_OF_FILE
if test 1057 -ne `wc -c <'top/UTIL.C'`; then
    echo shar: \"'top/UTIL.C'\" unpacked with wrong size!
fi
# end of 'top/UTIL.C'
fi
echo shar: End of archive 1 \(of 9\).
cp /dev/null ark1isdone
```

```
MISSING=""
for I in 1 2 3 4 5 6 7 8 9 ; do
    if test ! -f ark${I}isdone ; then
     MISSING="${MISSING} ${I}"
    fi
done
if test "${MISSING}" = "" ; then
    echo You have unpacked all 9 archives.
    rm -f ark[1-9]isdone ark[1-9][0-9]isdone
else
    echo You still need to unpack the following archives:
    echo "        " ${MISSING}
fi
##  End of shell archive.
exit 0


From -3239490533276823625
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,9947e9c5e9ea4f36,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-24 18:47:32 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!uwm.edu!cs.utexas.edu!
tut.cis.ohio-state.edu!gem.mps.ohio-state.edu!apple!sun-barr!newstop!sun!
grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i093: sozobon1.2 -- Update to Sozobon C compiler  part02/09
Keywords: shar, C
Message-ID: <34604@grapevine.uucp>
Date: 25 Oct 89 01:47:32 GMT
Sender: news@grapevine.EBay
Reply-To: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Lines: 2166
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Posting-number: Volume 2, Issue 93
Archive-name: sozobon1.2/part02

#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of archive 2 (of 9)."
# Contents:  hcc/MAIN.C hcc/MD.C hcc/NODES.H top/FUNC.C top/INST.C
#   top/PEEP1.C
# Wrapped by koreth@panarthea on Tue Oct 24 18:40:44 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'hcc/MAIN.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/MAIN.C'\"
```

```
else
echo shar: Extracting \"'hcc/MAIN.C'\" \(8368 characters\)
sed "s/^X//" >'hcc/MAIN.C' <<'END_OF_FILE'
X/* Copyright (c) 1988,1989 by Sozobon, Limited.  Author: Johann Ruegg
X * 
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  main.c
X *
X *  Main routine, error handling, keyword lookup.
X */
X
X#include <stdio.h>
X#include "param.h"
X#include "nodes.h"
X#include "tok.h"
X
Xint lineno;
Xint nmerrors;
Xint oflags[26];
Xint xflags[26];
Xint pflag = 0;             /* enable profiling */
Xstatic int anydebug;
X#define debug oflags['z'-'a']
X
XFILE *input;
XFILE *output;
X#if CC68
XFILE *fopenb();
X#define fopen fopenb
X#endif
Xchar *inname;
X
X#if NEEDBUF
Xchar my_ibuf[BUFSIZ];
X#endif
X
X#ifdef MINIX
X#define strchr    index
X#endif
X
XNODEP cur;
X
X#define MAXPREDEF 20
X
Xstruct def {
X    char *dname, *dval;
```

```
X} defines[MAXPREDEF] = {
X    {"MC68000"},
X    {"mc68000"},
X    {"SOZOBON"},
X#ifdef FOR_AMIGA
X    {"AMIGA"},
X    {"AMIGADOS"},
X#else
X    {"ATARI_ST"},
X#ifdef MINIX
X    {"MINIX"},
X#else
X    {"TOS"},
X#endif
X#endif
X};
Xstatic int npred = 4;
X
X#ifdef MINIX
Xchar    tmpdir[128] = ".";      /* where the output goes */
X#endif
X
Xmain(argc, argv)
Xchar **argv;
X{
X    char *p, *getenv();
X    int shownames;
X    int i;
X
X    if (sizeof(NODE) & 3) {
X        printf("sizeof NODE not mult of 4\n");
X        exit(1);
X    }
X
X    /*
X     * Parse the INCLUDE environment variable, if present.
X     */
X    if ((p = getenv("INCLUDE")) != NULL)
X        doincl(p);
X
X    shownames = 0;
X#ifndef FOR_AMIGA
X    if (isatty(0)) {
X#ifndef MINIX
X        write(1, "\33v", 2);
X#endif
X        setbuf(stdout, NULL);
X    }
X#endif
X/* put author here */
X    while (argc-- > 1) {
X        argv++;
X        if (argv[0][0] == '-')
```

```
X             doopt(&argv[0][1]);
X#if CC68
X         else if (argv[0][0] == '+') {
X             upstr(&argv[0][1]);
X             doopt(&argv[0][1]);
X         }
X#endif
X         else {
X             if (argc > 1 || shownames) {
X                  shownames++;
X                  printf("%s:\n", argv[0]);
X             }
X             if (input != NULL)
X                  fclose(input);
X             input = fopen(argv[0], ROPEN);
X             if (input == NULL) {
X                  printf("Cant open %s\n", argv[0]);
X                  exit(1);
X             }
X#if NEEDBUF
X             setbuf(input, my_ibuf);
X#endif
X             inname = argv[0];
X             dofile();
X         }
X    }
X    if (input == NULL) {
X         input = stdin;
X         output = stdout;
X         inname = "<STDIN>";
X         dofile();
X    }
X    exit(0);
X}
X
Xadddef(s)
Xchar *s;
X{
X    char *as, *strchr();
X
X    if (npred >= MAXPREDEF) {
X         warn("too many -D 's");
X         return;
X    }
X    if ((as = strchr(s,'=')) != NULL)
X         *as++ = 0;
X    else
X         as = NULL;
X    defines[npred].dname = s;
X    defines[npred].dval = as;
X    npred++;
X}
X
```

```
Xsubdef(s)
Xchar *s;
X{
X    int i;
X
X    for (i=0; i<npred; i++)
X         if (strcmp(s, defines[i].dname) == 0)
X              goto found;
X    return;
Xfound:
X    while (i < npred) {
X         defines[i] = defines[i+1];
X         i++;
X    }
X    npred--;
X}
X
Xdodefs()
X{
X    int i;
X    struct def *p;
X
X    /*
X     * Define the "built-in" macros
X     */
X    p = defines;
X    for (i=0; i < npred; i++,p++)
X         optdef(p->dname, p->dval ? p->dval : "1");
X}
X
Xdoincl(s)
Xchar    *s;
X{
X    char *malloc(), *strcpy();
X    char buf[256];
X    char dir[128];
X    register char *p;
X    char c;
X
X    strcpy(buf, s);
X
X    /*
X     * Convert ',' and ';' to nulls
X     */
X    for (p=buf; *p != '\0' ;p++)
X         if (*p == ',' || *p == ';')
X              *p = '\0';
X    p[1] = '\0';           /* double null terminated */
X
X    /*
X     * Grab each directory, make sure it ends with a slash
X     * and add it to the directory list.
X     */
```

```
X    for (p=buf; *p != '\0' ;p++) {
X         strcpy(dir, p);
X         c = dir[strlen(dir)-1];
X#ifndef FOR_AMIGA
X#ifdef MINIX
X         if (c != '/')
X              strcat(dir, "/");
X#else
X         if (c != '\\')
X              strcat(dir, "\\");
X#endif
X#else
X         if (c != '/' && c != ':')
X              strcat(dir, "/");
X#endif
X
X         optincl( strcpy(malloc((unsigned) (strlen(dir) + 1)), dir) );
X
X         while (*p != '\0')
X              p++;
X    }
X}
X
Xdofile()
X{
X    extern int nodesmade, nodesavail;
X    char *scopy();
X    extern NODEP deflist[], symtab[], tagtab;
X    extern NODEP strsave;
X    extern int level;
X    int i;
X
X    out_start(inname);
X    inname = scopy(inname);
X    lineno = 1;
X    nmerrors = 0;
X    dodefs();
X    advnode();
X
X    level = 0;
X    program();
X    dumpstrs(strsave);
X#ifdef OUT_AZ
X    xrefs();
X#endif
X
X    out_end();
X    if (cur && cur->e_token == EOFTOK)
X         freenode(cur);
X    sfree(inname);
X    for (i=0; i<NHASH; i++) {
X         if (debug>1 && deflist[i]) {
X              printf("defines[%d]", i);
```

```
X             printlist(deflist[i]);
X        }
X        freenode(deflist[i]);
X        deflist[i] = NULL;
X        if (debug && symtab[i]) {
X             printf("gsyms[%d]", i);
X             printlist(symtab[i]);
X        }
X        freenode(symtab[i]);
X        symtab[i] = NULL;
X    }
X    if (debug) {
X         printf("structs");
X         printlist(tagtab);
X    }
X    freenode(tagtab);
X    tagtab = NULL;
X    freenode(strsave);
X    strsave = NULL;
X    if (nmerrors) {
X         printf("%d errors\n", nmerrors);
X         exit(1);
X    }
X    if (nodesmade != nodesavail) {
X         printf("lost %d nodes!!!\n", nodesmade-nodesavail);
X         exit(1);
X    }
X/*
X    printf("Space = %ldK\n", ((long)nodesavail*sizeof(NODE))/1024);
X*/
X}
X
Xstatic   char Version[] =
X"hcc: version 1.20  Copyright (c) 1988,1989 by Sozobon, Limited.";
X
Xdoopt(s)
Xchar *s;
X{
X    register char c;
X
X    while ((c = *s++)) {
X#ifdef  DEBUG
X        if (c >= 'a' && c <='z') {
X             oflags[c-'a']++;
X             anydebug++;
X        } else
X#endif
X        if (c >= 'A' && c <= 'Z') {
X             switch (c) {
X             case 'D':
X                  adddef(s);
X                  return;
X             case 'U':
```

```
X                  subdef(s);
X                  return;
X              case 'I':
X                  doincl(s);
X                  return;
X              case 'P':
X                  pflag = 1;
X                  continue;
X              case 'V':
X                  printf("%s\n", Version);
X                  continue;
X#ifdef MINIX
X              case 'T':
X                  strcpy(tmpdir, s);
X                  if (tmpdir[strlen(tmpdir)-1] == '/')
X                      tmpdir[strlen(tmpdir)-1] = '\0';
X                  return;
X#endif
X              }
X#ifdef  DEBUG
X              xflags[c-'A']++;
X              anydebug++;
X#endif
X          }
X      }
X}
X
Xerrors(s,t)
Xchar *s, *t;
X{
X    optnl();
X    printf("error in %s on line %d: %s %s\n", inname, lineno, s,t);
X    nmerrors++;
X}
X
Xerrorn(s,np)
Xchar *s;
XNODE *np;
X{
X    optnl();
X    printf("error in %s on line %d: %s ", inname, lineno, s);
X    put_nnm(np);
X    putchar('\n');
X    nmerrors++;
X}
X
Xerror(s)
Xchar *s;
X{
X    optnl();
X    printf("error in %s on line %d: %s\n", inname, lineno, s);
X    nmerrors++;
X}
```

```
X
Xwarns(s,t)
Xchar *s, *t;
X{
X    optnl();
X    printf("warning in %s on line %d: %s %s\n", inname, lineno, s,t);
X}
X
Xwarnn(s,np)
Xchar *s;
XNODE *np;
X{
X    optnl();
X    printf("warning in %s on line %d: %s ", inname, lineno, s);
X    put_nnm(np);
X    putchar('\n');
X}
X
Xwarn(s)
Xchar *s;
X{
X    optnl();
X    printf("warning in %s on line %d: %s\n", inname, lineno, s);
X}
X
Xfatals(s,t)
Xchar *s, *t;
X{
X    optnl();
X    printf("fatal error in %s on line %d: %s %s\n", inname, lineno, s,t);
X    exit(1);
X}
X
Xfataln(s,np)
Xchar *s;
XNODE *np;
X{
X    optnl();
X    printf("fatal error in %s on line %d: %s ", inname, lineno, s);
X    put_nnm(np);
X    putchar('\n');
X    exit(1);
X}
X
Xfatal(s)
Xchar *s;
X{
X    optnl();
X    printf("fatal error in %s on line %d: %s\n", inname, lineno, s);
X    exit(1);
X}
X
Xstatic
```

```
Xoptnl()
X{
X    if (anydebug)
X         putchar('\n');
X}
X
Xstruct kwtbl {
X    char *name;
X    int  kwval;
X    int  kflags;
X} kwtab[] = {
X    /* must be sorted */
X    {"asm", K_ASM},
X    {"auto", K_AUTO},
X    {"break", K_BREAK},
X    {"case", K_CASE},
X    {"char", K_CHAR},
X    {"continue", K_CONTINUE},
X    {"default", K_DEFAULT},
X    {"do", K_DO},
X    {"double", K_DOUBLE},
X    {"else", K_ELSE},
X    {"enum", K_ENUM},
X    {"extern", K_EXTERN},
X    {"float", K_FLOAT},
X    {"for", K_FOR},
X    {"goto", K_GOTO},
X    {"if", K_IF},
X    {"int", K_INT},
X    {"long", K_LONG},
X    {"register", K_REGISTER},
X    {"return", K_RETURN},
X    {"short", K_SHORT},
X    {"sizeof", K_SIZEOF},
X    {"static", K_STATIC},
X    {"struct", K_STRUCT},
X    {"switch", K_SWITCH},
X    {"typedef", K_TYPEDEF},
X    {"union", K_UNION},
X    {"unsigned", K_UNSIGNED},
X    {"void", K_VOID},
X    {"while", K_WHILE},
X
X    {0,0}
X};
X
X#define FIRST_C   'a'
X#define LAST_C    'z'
Xstruct kwtbl *kwstart[LAST_C-FIRST_C+1];
X
Xkw_init()
X{
X    register struct kwtbl *p;
```

```
X    register c;
X
X    for (p=kwtab; p->name; p++) {
X         c = p->name[0];
X         if (kwstart[c-FIRST_C] == 0)
X              kwstart[c-FIRST_C] = p;
X    }
X}
X
Xkw_tok(tp)
XNODE *tp;
X{
X    register struct kwtbl *kp;
X    register char *nm;
X    register i;
X    static first = 0;
X
X    nm = tp->n_name;
X    if (first == 0) {
X         kw_init();
X         first = 1;
X    }
X    i = nm[0];
X    if (i < FIRST_C || i > LAST_C)
X         return;
X    kp = kwstart[i-FIRST_C];
X    if (kp)
X    for (; kp->name; kp++) {
X         i = strcmp(nm, kp->name);
X         if (i == 0) {
X              tp->e_token = kp->kwval;
X              tp->e_flags = kp->kflags;
X              return;
X         } else if (i < 0)
X              return;
X    }
X}
X
X#if CC68
X/* fix args since stupid lib makes all lower case */
Xupstr(s)
Xchar *s;
X{
X    while (*s) {
X         if (*s >= 'a' && *s <= 'z')
X              *s += 'A'-'a';
X         s++;
X    }
X}
Xdownstr(s)
Xchar *s;
X{
X    while (*s) {
```

```
X         if (*s >= 'A' && *s <= 'Z')
X             *s -= 'A'-'a';
X         s++;
X    }
X}
X#endif
END_OF_FILE
if test 8368 -ne `wc -c <'hcc/MAIN.C'`; then
    echo shar: \"'hcc/MAIN.C'\" unpacked with wrong size!
fi
# end of 'hcc/MAIN.C'
fi
if test -f 'hcc/MD.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/MD.C'\"
else
echo shar: Extracting \"'hcc/MD.C'\" \(7524 characters\)
sed "s/^X//" >'hcc/MD.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  md.c
X *
X *  Machine dependant parts of first pass (parse)
X *  Also type checking subroutines.
X */
X
X#include <stdio.h>
X#include "param.h"
X#include "tok.h"
X#include "nodes.h"
X#include "cookie.h"
X
XNODEP bas_type();
X
Xint adjtab[] = {
X    K_INT,        /* none */
X    K_SHORT,  /* short */
X    K_LONG,       /* long */
X    0,        /* short long */
X    K_UNSIGNED,   /* unsigned */
X    K_UNSIGNED,   /* unsigned short */
X    T_ULONG,  /* unsigned long */
X    0,        /* unsigned short long */
X};
X
Xadj_type(old, adj)
```

```
X{
X    int rv;
X
X    switch (old) {
X    case K_CHAR:
X         if (adj & SAW_UNS)
X              return T_UCHAR;
X         break;
X    case K_INT:
X         rv = adjtab[adj];
X         if (rv == 0) {
X              error("bad type spec");
X              return K_INT;
X         }
X         return rv;
X    case K_FLOAT:
X         if (adj & SAW_LONG)
X              return K_DOUBLE;
X         break;
X    }
X    return old;
X}
X
X/* given ICON value, and flags SEE_L,SEE_U
X    determine final type */
Xicon_ty(tp)
XNODE *tp;
X{
X    int flags;
X    long val;
X    int islong, isuns;
X
X    flags = tp->e_flags;
X    val = tp->e_ival;
X
X    islong = (flags & SEE_L);
X    isuns = (flags & SEE_U);
X
X    if (islong && isuns)
X         return T_ULONG;
X    if (islong || islongv(val))
X         return K_LONG;
X    if (isuns)
X         return K_UNSIGNED;
X    return isintv((int)val) ? K_INT : K_CHAR;
X}
X
Xisintv(i)
X{
X    if (i > 0x7f || i < -0x80)
X         return 1;
X    return 0;
X}
```

```
X
Xislongv(l)
Xlong l;
X{
X#ifndef NOLONGS
X#define  HIBITS   0xffff0000L
X
X    if ((l & HIBITS) == 0)      /* upper 16 bits zero */
X         return 0;
X
X    if ((l & HIBITS) == HIBITS) {    /* upper bits all on */
X         if (l & 0x8000L)
X              return 0; /* upper bits aren't significant */
X         else
X              return 1;
X    }
X    return 1;
X#else
X    return 0;
X#endif
X}
X
Xmkint(l)
Xlong l;
X{
X    return l;
X}
X
Xlc_reg(rp, xp)
Xint *rp;
XNODE *xp;
X{
X    switch (xp->n_tptr->t_token) {
X    case STAR:
X         return al_areg(rp,xp);
X    case K_CHAR:
X    case T_UCHAR:
X    case T_ULONG:
X    case K_INT:
X    case K_UNSIGNED:
X    case K_LONG:
X         return al_dreg(rp,xp);
X    default:
X         return 0;
X    }
X}
X
Xal_areg(rp,xp)
Xint *rp;
XNODEP xp;
X{
X    register rmask, n;
X
```

```
X    rmask = *rp;
X    for (n=ARV_START; n<=ARV_END; n++)
X         if ((rmask & (1<<n)) == 0) {
X              xp->e_rno = n;
X              *rp |= (1<<n);
X              return 1;
X         }
X    return 0;
X}
X
Xal_dreg(rp,xp)
Xint *rp;
XNODEP xp;
X{
X    register rmask, n;
X
X    rmask = *rp;
X    for (n=DRV_START; n<=DRV_END; n++)
X         if ((rmask & (1<<n)) == 0) {
X              xp->e_rno = n;
X              *rp |= (1<<n);
X              return 1;
X         }
X    return 0;
X}
X
Xlong
Xarg_size(sz,np)
Xlong sz;
XNODEP np;
X{
X    np->e_offs = 0;
X
X    switch (np->n_tptr->t_token) {
X    case '[':
X         printf("GAK! array arg ");
X         return SIZE_P;
X    case K_CHAR:
X    case T_UCHAR:
X         np->e_offs = SIZE_I - SIZE_C;
X         return SIZE_I;
X#if SIZE_I != SIZE_S
X    case K_SHORT:
X         np->e_offs = SIZE_I - SIZE_S;
X         return SIZE_I;
X#endif
X    default:
X         if (sz & 1)
X              sz++;
X         return sz;
X    }
X}
X
```

```
Xmustlval(np)
XNODEP np;
X{
X    switch (np->e_token) {
X    case ID:
X    case STAR:
X    case '.':
X         break;
X    default:
X         errorn("not lvalue", np);
X         return 1;
X    }
X    return 0;
X}
X
Xmustty(np, flags)
XNODEP np;
X{
X    switch (np->n_tptr->t_token) {
X    case STAR:
X         if (flags & R_POINTER)
X              return 0;
X         error("pointer not allowed");
X         return 1;
X    case K_STRUCT:
X    case K_UNION:
X         if (flags & R_STRUCT)
X              return 0;
X         error("struct/union not allowed");
X         return 1;
X    case K_CHAR:
X    case K_SHORT:
X    case K_INT:
X    case K_UNSIGNED:
X    case K_LONG:
X    case T_UCHAR:
X    case T_ULONG:
X         if (flags & R_INTEGRAL)
X              return 0;
X         error("integral not allowed");
X         return 1;
X    case K_FLOAT:
X    case K_DOUBLE:
X         if (flags & R_FLOATING)
X              return 0;
X         error("floating not allowed");
X         return 1;
X    default:
X         error("bad type");
X         return 1;
X    }
X    return 0;
X}
```

```
X
XNODEP
Xfuncty(np)
XNODEP np;
X{
X    int lt;
X
X    lt = np->n_tptr->t_token;
X    if (lt != K_VOID)
X         mustty(np, R_ASSN);
X    switch (lt) {
X    case STAR:
X    case K_STRUCT:
X    case K_UNION:
X         return np->n_tptr;
X    }
X    lt = widen(lt);
X    return bas_type(lt);
X}
X
XNODEP
Xnormalty(lp, rp)
XNODEP lp, rp;
X{
X    /* already checked types are R_ARITH */
X    /* rp may be NULL */
X    int lt, rt, rett;
X
X    lt = lp->n_tptr->t_token;
X    if (rp)
X         rt = rp->n_tptr->t_token;
X    else
X         rt = K_INT;
X    rett = maxt(widen(lt), widen(rt));
X    return bas_type(rett);
X}
X
Xasn_chk(ltp, rp)
XNODEP ltp, rp;
X{
X
X    switch (ltp->t_token) {
X    case K_STRUCT:
X    case K_UNION:
X         if (same_type(ltp, rp->n_tptr) == 0)
X              error("bad struct assign");
X         return;
X    case STAR:
X         if (mayzero(rp))
X              return;
X         if (mustty(rp, R_POINTER))
X              return;
X         if (same_type(ltp->n_tptr, rp->n_tptr->n_tptr)
```

```
X             == 0)
X             warn("pointer types mismatch");
X        return;
X    default:
X        if (mustty(rp, R_ARITH))
X             return;
X    }
X}
X
Xchkcmp(np)
XNODEP np;
X{
X    /* already checked types are R_SCALAR */
X    int lt, rt;
X    NODEP lp = np->n_left, rp = np->n_right;
X
X    lt = lp->n_tptr->t_token;
X    lt = (lt == STAR);
X    rt = rp->n_tptr->t_token;
X    rt = (rt == STAR);
X    if (lt && rt) {        /* ptr cmp ptr */
X        if (same_type(lp->n_tptr, rp->n_tptr) == 0) {
X             warn("cmp of diff ptrs");
X        }
X    } else if (lt) {   /* ptr cmp intg */
X        mustzero(rp);
X    } else if (rt) {   /* intg +-[ ptr */
X        mustzero(lp);
X    } /* else both ARITH */
X}
X
XNODEP
Xcolonty(np)
XNODEP np;
X{
X    /* already checked types are R_SCALAR */
X    int lt, rt;
X    NODEP lp = np->n_left, rp = np->n_right;
X
X    lt = lp->n_tptr->t_token;
X    lt = (lt == STAR);
X    rt = rp->n_tptr->t_token;
X    rt = (rt == STAR);
X    if (lt && rt) {        /* ptr : ptr */
X        warn(": diff ptrs");
X        return lp->n_tptr;
X    } else if (lt) {   /* ptr : intg */
X        mustzero(rp);
X        return lp->n_tptr;
X    } else if (rt) {
X        mustzero(lp);
X        return rp->n_tptr;
X    } else
```

```
X        return normalty(lp, rp);
X}
X
XNODEP
Xaddty(np)
XNODEP np;
X{
X    /* already checked types are R_SCALAR */
X    /* op is '+' or '-' or '+=' or '-=' or '[' */
X    int oop = np->e_token;
X    int op;
X    int lt, rt;
X    NODEP lp = np->n_left, rp = np->n_right;
X
X    op = oop;
X    if (isassign(op))
X        op -= ASSIGN 0;
X    lt = lp->n_tptr->t_token;
X    lt = (lt == STAR);
X    rt = rp->n_tptr->t_token;
X    rt = (rt == STAR);
X    if (lt && rt) {         /* ptr - ptr */
X        if (oop != '-' || same_type(lp->n_tptr, rp->n_tptr) == 0) {
X            error("bad +/-");
X            return lp->n_tptr;
X        }
X        np->e_token = PTRDIFF;
X        np->e_offs = lp->n_tptr->n_tptr->t_size;
X        return bas_type(K_INT);
X    } else if (lt) {   /* ptr +-[ intg */
Xpandi:
X        mustty(rp, R_INTEGRAL);
X        np->e_offs = lp->n_tptr->n_tptr->t_size;
X        if (op == '+')
X            np->e_token += PTRADD-'+';
X        else if (op == '-')
X            np->e_token += PTRSUB-'-';
X        return lp->n_tptr;
X    } else if (rt) {   /* intg +-[ ptr */
X        if (isassign(oop) || op == '-') {
X            error("illegal int op ptr");
X            return bas_type(K_INT);
X        }
X        /* switch sides so intg is on right */
X        np->n_left = rp;
X        np->n_right = lp;
X        lp = rp;
X        rp = np->n_right;
X        goto pandi;
X    } else {      /* intg +- intg */
X        return normalty(lp, rp);
X    }
X}
```

```
X
Xmustzero(np)
XNODEP np;
X{
X    if (np->e_token == ICON && np->e_ival == 0) {
X        return;
X    }
X    error("bad ':' combination");
X}
X
Xmayzero(np)
XNODEP np;
X{
X    if (np->e_token == ICON && np->e_ival == 0) {
X        return 1;
X    }
X    return 0;
X}
X
Xwiden(ty)
X{
X    switch (ty) {
X    case K_CHAR:
X    case T_UCHAR:
X        return K_INT;
X    case K_SHORT:
X        return K_INT;
X    case K_FLOAT:
X        return K_DOUBLE;
X    default:
X        return ty;
X    }
X}
X
Xint pri_t[] = {
X    1, 6,         /* uchar, ulong */
X    5,2,4,3,0,    /* long, short, uns, int, char */
X    7,8,9    /* float, double, void */
X};
X
Xmaxt(t1, t2)
X{
X    extern nmerrors;
X
X    if (nmerrors)
X        return K_INT;
X    if (pri_t[t1-FIRST_BAS] > pri_t[t2-FIRST_BAS])
X        return t1;
X    return t2;
X}
END_OF_FILE
if test 7524 -ne `wc -c <'hcc/MD.C'`; then
    echo shar: \"'hcc/MD.C'\" unpacked with wrong size!
```

```
fi
# end of 'hcc/MD.C'
fi
if test -f 'hcc/NODES.H' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/NODES.H'\"
else
echo shar: Extracting \"'hcc/NODES.H'\" \(4451 characters\)
sed "s/^X//" >'hcc/NODES.H' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X * 
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  nodes.h
X */
X
X/*
X *  stuff common to all nodes
X */
X#define COMMON    int cflags;   \
X         int fill;\
X         union node *left;  \
X         union node *right; \
X         union node *tptr;  \
X         union node *nm_ext;    \
X         char cname[NMSIZE]
X
X#define n_flags   e.cflags
X#define n_left    e.left
X#define n_next    e.left
X#define n_right e.right
X#define n_tptr    e.tptr
X#define n_nmx     e.nm_ext
X#define n_name    e.cname
X
X/*
X *  expression (and symbol table) node
X */
Xstruct enode {
X    COMMON;
X    int  token;        /* must be same place as tnode */
X    int  eflags;
X    char etype;        /* type of node */
X    char sc;
X    char eprec;
X    char rno;
X    union {
X         long vival;
```

```
X        long voffs;
X        double   vfval;
X    } vu;
X#ifndef NOFIELDS
X    unsigned  fldw:6, fldof:6;   /* use fields just so
X                              we know fields work */
X#else
X    char fldw, fldof;       /* use if cant do fields */
X#endif
X};
X
X#define e_token   e.token
X#define e_flags   e.eflags
X#define e_prec    e.eprec
X#define e_rno     e.rno
X#define e_type    e.etype
X#define e_ival    e.vu.vival
X#define e_offs    e.vu.voffs
X#define e_fval    e.vu.vfval
X#define e_sc e.sc
X#define e_fldw    e.fldw
X#define e_fldo    e.fldof
X
X/* for e_flags values, see tok.h */
X
X/* values for e_type */
X
X#define E_LEAF    0    /* no descendants */
X#define E_UNARY 1 /* left node is expr, no right node */
X#define E_BIN     2    /* left and right are expr */
X#define E_SPEC    3    /* special '(', '[', '.', '->', ... */
X
X/*
X * code generation node
X */
Xstruct gnode {
X    COMMON;
X    int  token;
X    int  eflags;
X    char etype;
X    char sc;
X/* all of above fields must match first fields in enode! */
X
X    char needs;         /* registers needed */
X    char grno;          /* register used in ret value */
X    char basety;        /* type FLOAT, UNS, INT or AGREG */
X    char basesz;        /* size 1,2,4 or 3 -> see bsize */
X    char gr1, gr2;
X    char *betwc;        /* code for between L and R */
X    long goffs;         /* offsets for OREG, ONAME */
X    union gu {
X         long bsize;         /* AGREG size or misc. */
X         struct {
```

```
X#ifndef NOFIELDS
X#ifdef AZ_HOST
X             unsigned gfldw:6, gfldo:6;  /* BUG! */
X#else
X             int  gfldw:6, gfldo:6;  /* field info */
X#endif
X#else
X             char gfldw, gfldo; /* use if no fields */
X#endif
X         } gfl;
X    } gu;
X};
X
X#define g_token   g.token
X#define g_flags   g.eflags
X#define g_type    g.etype
X#define g_sc g.sc
X#define g_needs g.needs
X#define g_rno     g.grno
X#define g_offs    g.goffs
X#define g_betw    g.betwc
X#define g_ty g.basety
X#define g_sz g.basesz
X#define g_code    g.tptr
X#define g_bsize   g.gu.bsize
X#define g_fldw    g.gu.gfl.gfldw
X#define g_fldo    g.gu.gfl.gfldo
X#define g_r1 g.gr1
X#define g_r2 g.gr2
X
X/* types of operands -- ordered in cast strength order */
X#define ET_S 1    /* signed integer */
X#define ET_U 2    /* unsigned integer */
X#define ET_F 3    /* float or double */
X#define ET_A 4    /* aggregate */
X
X/*
X *  type list node
X */
Xstruct tnode {
X    COMMON;
X    int  token;        /* must be same place as enode */
X    int  tflags;
X    char aln;     /* alignment needed */
X    long tsize;
X};
X
X#define t_token   t.token
X#define t_flags   t.tflags
X#define t_size    t.tsize
X#define t_aln     t.aln
X
X/*
```

```
X *  name extension node
X */
Xstruct nmext {
X    COMMON;
X    char nmx[NMXSIZE-NMSIZE];   /* name extension (with name)*/
X};
X
X#define x_nm x.nmx
X
X/*
X *  block info node
X */
Xstruct bnode {
X    COMMON;
X    union node *syms;
X    union node *tags;
X    int  regs;         /* reg alloc mask */
X    long lsize;        /* size of locals */
X    int  tmsize;       /* max tmps used for 1 expr */
X};
X
X#define b_syms    b.syms
X#define b_tags    b.tags
X#define b_regs    b.regs
X#define b_size    b.lsize
X#define b_tsize b.tmsize
X
X/*
X *  node to hold case for switch generation
X */
Xstruct cnode {
X    COMMON;
X    int  c_value; /* value for case */
X    int  c_label; /* case label or label label */
X    int  c_def;        /* label defined */
X};
X
X#define c_defined c.c_def
X#define c_casev c.c_value
X#define c_casel c.c_label
X
Xunion node {
X    struct enode e;
X    struct tnode t;
X    struct nmext x;
X    struct bnode b;
X    struct cnode c;
X    struct gnode g;
X};
X
Xtypedef union node NODE;
Xtypedef NODE *NODEP;
X
```

```
X/* defines for n_flags */
X
X#define N_BRKPR   1    /* break printnode recursion */
X#define N_COPYT   2    /* tptr is a copy */
X#define N_ISFREE 4     /* node is on free list (error check) */
X
XNODEP allocnode();
END_OF_FILE
if test 4451 -ne `wc -c <'hcc/NODES.H'`; then
    echo shar: \"'hcc/NODES.H'\" unpacked with wrong size!
fi
# end of 'hcc/NODES.H'
fi
if test -f 'top/FUNC.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/FUNC.C'\"
else
echo shar: Extracting \"'top/FUNC.C'\" \(4582 characters\)
sed "s/^X//" >'top/FUNC.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X#include "top.h"
X
XBLOCK   *fhead;       /* head of the current function */
X
X/*
X * dofunc() - process one function
X *
X * Returns FALSE on end of file
X */
Xbool
Xdofunc()
X{
X    BLOCK    *getfunc();
X
X    clrvar();
X
X#ifdef  DEBUG
X    if (debug)
X        fprintf(stderr, "dofunc() - calling getfunc()\n");
X#endif
X    if ((fhead = getfunc()) == NULL)
X        return FALSE;
X
X    /*
X     * Process the function we just read
```

```
X     */
X    bopt(fhead);       /* perform branch optimization */
X
X    if (do_regs)
X         setreg(fhead);     /* try to assign locals to registers */
X
X    if (do_peep) {
X         rhealth(fhead, TRUE);  /* live/dead register analysis */
X         peep(fhead);       /* peephole optimizations */
X    }
X
X    /*
X     * Now dump out the modified tree
X     */
X#ifdef  DEBUG
X    if (debug)
X         fprintf(stderr, "dofunc() - calling putfunc()\n");
X#endif
X    putfunc(fhead);
X
X    freesym();         /* free the symbol table */
X
X    return TRUE;
X}
X
Xstatic   bool saw_eof = FALSE;
X
X/*
X * getfunc() - get a function and return a pointer to its starting block
X *
X * Returns NULL on end of file.
X */
XBLOCK *
Xgetfunc()
X{
X    register BLOCK     *head;   /* starting block for this function */
X    register BLOCK     *cb; /* the block we're currently reading */
X    register BLOCK     *ob; /* the last block we read */
X
X    if (saw_eof)
X         return NULL;
X
X    head = NULL;
X
X    /*
X     * Starting a global function
X     */
X    if (strcmp(t_op, ".globl") == 0) {
X         /*
X          * Enter the symbol and mark it global.
X          */
X         head = mksym(t_arg);
X         head->flags |= B_GLOBAL;
```

```
X
X         readline();
X    }
X
X    ob = NULL;
X
X    for (;;) {
X         if (ob == NULL) {
X              if (t_lab[0] != '_') {
X                   fprintf(stderr, "top: expected function label\n");
X                   exit(1);
X              }
X              if (head == NULL)
X                   head = mksym(t_lab);
X
X         } else if (t_lab[0] == '\0') {
X              fprintf(stderr, "top: expected block label\n");
X              exit(1);
X         }
X
X         if ((cb = getsym(t_lab)) == NULL)
X              cb = mksym(t_lab);
X
X         /*
X          * The last block falls through to this one.
X          */
X         if (ob != NULL) {
X              ob->chain = cb;
X              ob->next = cb;
X              ob->bfall = cb;
X         }
X
X         t_lab[0] = '\0';
X
X         /*
X          * Now read lines until we hit a new block or another
X          * function.
X          */
X         for (;;) {
X              /*
X               * If we see a global, we're done with the function
X               */
X              if (strcmp(t_op, ".globl") == 0)
X                   return head;
X              /*
X               * If we see a function label, we're done too.
X               */
X              if (t_lab[0] == '_')
X                   return head;
X              /*
X               * If we see any other label, we're done with the block.
X               */
X              if (t_lab[0])
```

```
X                   break;
X
X              addinst(cb, t_op, t_arg);
X
X              /*
X               * If we're at EOF, note that we've hit the end of
X               * file, but return the function we just read.
X               */
X              if (!readline()) {
X                   saw_eof = TRUE;
X                   return head;
X              }
X         }
X         ob = cb;
X    }
X}
X
X/*
X * putfunc(sb) - print out the function starting at block 'sb'
X *
X * The 'next' pointers determine the order in which things are placed
X * in the file. Branch instructions have been removed so they need to
X * be replaced here on output. Conditional branches are generated if
X * indicated (by non-null 'bcond'). Unconditional branches are generated
X * at the end of a block if it's "fall through" block isn't going to
X * be the next thing in the file.
X */
Xputfunc(sb)
Xregister BLOCK    *sb;
X{
X    register BLOCK     *cb;
X    register INST *ci;
X
X    fprintf(ofp, "\t.text\n");
X
X    for (cb = sb; cb != NULL ;cb = cb->next) {
X         if (cb->flags & B_GLOBAL)
X              fprintf(ofp, "\t.globl\t%s\n", cb->name);
X
X         if (*cb->name == '_')
X              fprintf(ofp, "%s:\n", cb->name);
X
X         else if (cb->flags & B_LABEL)
X              fprintf(ofp, "%s:\n", cb->name);
X#ifdef   DEBUG
X         if (debug) {
X              fprintf(ofp, "*\n");
X              fprintf(ofp, "* %s, ref:%04x  set:%04x\n",
X                   cb->name, cb->rref, cb->rset);
X              fprintf(ofp, "*\n");
X         }
X#endif
X
```

```
X         for (ci = cb->first; ci != NULL ;ci = ci->next)
X             putinst(ci);
X         /*
X          * If there's a conditional branch, put out the
X          * appropriate instruction for it.
X          */
X         if (cb->bcond != NULL && cb->bcode >= 0)
X             fprintf(ofp, "\t%s\t%s\n",
X                  opnames[cb->bcode], cb->bcond->name);
X         /*
X          * If there's a "fall through" label, and the destination
X          * block doesn't come next, put out a branch.
X          */
X         if (cb->bfall != NULL && cb->bfall != cb->next) {
X             s_badd++;
X             fprintf(ofp, "\tbra\t%s\n", cb->bfall->name);
X         }
X    }
X}
END_OF_FILE
if test 4582 -ne `wc -c <'top/FUNC.C'`; then
    echo shar: \"'top/FUNC.C'\" unpacked with wrong size!
fi
# end of 'top/FUNC.C'
fi
if test -f 'top/INST.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/INST.C'\"
else
echo shar: Extracting \"'top/INST.C'\" \(7882 characters\)
sed "s/^X//" >'top/INST.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X
X/*
X * Routines dealing with the parsing and output of instructions.
X */
X
X#include "top.h"
X
Xstatic  void getarg();
Xstatic  int  isreg();
X
X/*
X * addinst(bp, op, args) - add an instruction to block 'bp'
X */
```

```
Xvoid
Xaddinst(bp, op, args)
Xregister BLOCK    *bp;
Xchar     *op, *args;
X{
X    register INST *ni;
X    register int  i;
X    register char *s;
X    char *arg2 = "";
X
X    if (*op == '\0')   /* no instruction there */
X         return;
X
X    ni = (INST *) alloc(sizeof(INST));
X
X    ni->flags = 0;
X    ni->opcode = -1;
X    ni->next = NULL;
X    ni->prev = NULL;
X    ni->live = 0;
X    ni->rref = ni->rset = 0;
X
X    ni->src.areg = ni->dst.areg = 0;
X    ni->src.ireg = ni->dst.ireg = 0;
X    ni->src.disp = ni->dst.disp = 0;
X    ni->src.amode = ni->dst.amode = NONE;
X
X    /*
X     * Link into the block appropriately
X     */
X    if (bp->first == NULL) {
X         bp->first = bp->last = ni;
X    } else {
X         bp->last->next = ni;
X         ni->prev = bp->last;
X
X         bp->last = ni;
X    }
X
X    for (s = op; *s ;s++) {
X         /*
X          * Pseudo-ops start with a period, so the length
X          * specifier can't be the first character.
X          */
X         if (*s == '.' && s != op) { /* length specifier */
X              *s++ = '\0';
X              switch (*s) {
X              case 'b':
X                   ni->flags |= LENB;
X                   break;
X              case 'w':
X                   ni->flags |= LENW;
X                   break;
```

```
X             case 'l':
X                  ni->flags |= LENL;
X                  break;
X             default:
X                  fprintf(stderr, "Bad length spec '%c'\n", *s);
X                  exit(1);
X             }
X        }
X    }
X
X    for (i=0; opnames[i] ;i++) {
X         if (strcmp(op, opnames[i]) == 0) {
X              ni->opcode = i;
X              break;
X         }
X    }
X
X    if (ni->opcode < 0) {
X         fprintf(stderr, "Unknown op '%s'\n", op);
X         exit(1);
X    }
X
X    /*
X     * Look for the split between the first and second operands.
X     */
X    for (s = args; *s ;s++) {
X         /*
X          * skip chars in parens, since an operand split can't
X          * occur within.
X          */
X         if (*s == '(') {
X              while (*s != ')')
X                   s++;
X         }
X         if (*s == ',') {
X              *s++ = '\0';
X              arg2 = s;
X              break;
X         }
X    }
X
X    getarg(&ni->src, args);
X    getarg(&ni->dst, arg2);
X}
X
X/*
X * delinst(bp, ip) - delete instruction 'ip' in block 'bp'
X */
Xvoid
Xdelinst(bp, ip)
XBLOCK   *bp;
Xregister INST    *ip;
X{
```

```
X    register INST *pi, *ni; /* previous and next instructions */
X
X    pi = ip->prev;
X    ni = ip->next;
X
X    if (pi != NULL)
X         pi->next = ni;
X    else
X         bp->first = ni;
X
X    if (ni != NULL)
X         ni->prev = pi;
X    else
X         bp->last = pi;
X
X    /*
X     * Free space used by the instruction.
X     */
X    freeop(&ip->src);
X    freeop(&ip->dst);
X    free(ip);
X
X    s_idel++;
X}
X
X/*
X * getarg(op, s) - parse string 's' into the operand structure 'op'
X *
X * Hack alert!! The following code parses the operands only to the
X * extent needed by the optimizer. We're primarily interested in
X * details about addressing modes used, not in any expressions that
X * might be present. This code is highly tuned to the output of the
X * compiler.
X */
Xstatic  void
Xgetarg(op, s)
Xregister struct   opnd *op;
Xregister char     *s;
X{
X    extern   long atol();
X    register int  reg;
X    register char *p;
X
X    if (*s == '\0') {
X         op->amode = NONE;
X         return;
X    }
X
X    if (*s == '#') {                 /* immediate data */
X         op->amode = IMM;
X         s += 1;
X         if (isdigit(s[0]) || s[0] == '-')
X              op->disp  = atol(s);
```

```
X         else {
X              op->amode |= SYMB;
X              op->astr = strsave(s);
X         }
X         return;
X    } else if ((reg = isreg(s)) >= 0) {       /* reg. direct */
X         op->amode = REG;
X         op->areg = reg;
X    } else if (s[0] == '(' || (s[0] == '-' && s[1] == '(')) {
X         op->amode = REGI;
X         if (s[0] == '-') {
X              op->amode |= DEC;
X              s++;
X         }
X         s++;      /* skip the left paren */
X         if ((op->areg = isreg(s)) < 0) {
X              fprintf(stderr, "bad reg. '%s'\n", s);
X              exit(1);
X         }
X         s += 3;       /* skip the register and right paren */
X
X         if (s[0] == '+')
X              op->amode |= INC;
X    } else if (!isdigit(s[0]) && (s[0] != '-')) {
X         op->amode = ABS;
X         op->astr = strsave(s);
X    } else {
X         for (p=s; isdigit(*p) || *p == '-' ;p++)
X              ;
X         if (*p != '(') {
X              /*
X               * Must have been absolute, but with an
X               * address instead of a symbol.
X               */
X              op->amode = ABS;
X              op->astr = strsave(s);
X              return;
X         }
X         *p++ = '\0';
X         op->disp = atol(s);
X         s = p;
X         if (s[0] == 'p' && s[1] == 'c') {    /* PC relative */
X              if (s[2] == ')') {
X                   op->amode = PCD;
X                   return;
X              }
X              op->amode = PCDX;
X              op->ireg = isreg(s+3);
X              if (s[6] == 'l')
X                   op->amode |= XLONG;
X         } else if ((reg = isreg(s)) >= 0) {
X              op->areg = reg;
X              if (s[2] == ')') {
```

```
X                  op->amode = REGID;
X                  return;
X             }
X             op->amode = REGIDX;
X             op->ireg = isreg(s+3);
X             if (s[6] == 'l')
X                  op->amode |= XLONG;
X        } else {
X             fprintf(stderr, "bad reg. '%s' after disp\n", s);
X             exit(1);
X        }
X    }
X}
X
X/*
X * characters that can terminate a register name
X */
X#define  isterm(c) ((c) == '\0' || (c) == ')' || (c) == ',' || (c) == '.')
X
Xstatic   int
Xisreg(s)
Xregister char     *s;
X{
X    if (s[0] == 'd' && isdigit(s[1]) && isterm(s[2]))
X         return D0 + (s[1] - '0');
X    if (s[0] == 'a' && isdigit(s[1]) && isterm(s[2]))
X         return A0 + (s[1] - '0');
X    if (s[0] == 's' && s[1] == 'p' && isterm(s[2]))
X         return SP;
X
X    return -1;
X}
X
X
X/*
X * Routines for printing out instructions
X */
X
Xstatic   char *rstr();
Xstatic   void putop();
X
Xvoid
Xputinst(ip)
Xregister INST     *ip;
X{
X    char c;
X
X    fprintf(ofp, "\t%s", opnames[ip->opcode]);
X
X    switch (ip->flags) {
X    case LENB:
X         c = 'b';
X         break;
```

```
X    case LENW:
X         c = 'w';
X         break;
X    case LENL:
X         c = 'l';
X         break;
X    default:
X         c = '\0';
X         break;
X    }
X    if (c)
X         fprintf(ofp, ".%c", c);
X
X    if (ip->src.amode != NONE) {
X         fprintf(ofp, "\t");
X         putop(&ip->src);
X    }
X
X    if (ip->dst.amode != NONE) {
X         fprintf(ofp, ",");
X         putop(&ip->dst);
X    }
X#ifdef  DEBUG
X    if (debug)
X         fprintf(ofp, "\t\t* ref(%04x), set(%04x), live(%04x)",
X              reg_ref(ip), reg_set(ip), ip->live);
X#endif
X    fprintf(ofp, "\n");
X}
X
Xstatic  void
Xputop(op)
Xregister struct   opnd *op;
X{
X    switch (op->amode & MMASK) {
X    case NONE:
X         break;
X    case REG:
X         fprintf(ofp, "%s", rstr(op->areg));
X         break;
X    case IMM:
X         if (op->amode & SYMB)
X              fprintf(ofp, "#%s", op->astr);
X         else
X              fprintf(ofp, "#%ld", op->disp);
X         break;
X    case ABS:
X         fprintf(ofp, "%s", op->astr);
X         break;
X    case REGI:
X         if (op->amode & DEC)
X              fprintf(ofp, "-");
X         fprintf(ofp, "(%s)", rstr(op->areg));
```

```
X         if (op->amode & INC)
X              fprintf(ofp, "+");
X         break;
X    case REGID:
X         fprintf(ofp, "%ld(%s)", op->disp, rstr(op->areg));
X         break;
X    case REGIDX:
X         fprintf(ofp, "%ld(%s,", op->disp, rstr(op->areg));
X         fprintf(ofp, "%s.%c)", rstr(op->ireg),
X              (op->amode & XLONG) ? 'l' : 'w');
X         break;
X    case PCD:
X         fprintf(ofp, "%ld(pc)", op->disp);
X         break;
X    case PCDX:
X         fprintf(ofp, "%ld(pc,%s.%c)", op->disp, rstr(op->ireg),
X              (op->amode & XLONG) ? 'l' : 'w');
X         break;
X    default:
X         fprintf(stderr, "bad addr. mode in putop: %d\n", op->amode);
X         exit(1);
X    }
X}
X
Xstatic   char *
Xrstr(r)
Xregister char     r;
X{
X    static    char buf[3];
X
X    if (r == SP) {
X         buf[0] = 's';
X         buf[1] = 'p';
X    } else if (r >= A0 && r <= A6) {
X         buf[0] = 'a';
X         buf[1] = '0' + (r - A0);
X    } else {
X         buf[0] = 'd';
X         buf[1] = '0' + (r - D0);
X    }
X    buf[2] = '\0';
X
X    return buf;
X}
X
X/*
X * opeq(op1, op2) - test equality of the two instruction operands
X */
Xbool
Xopeq(op1, op2)
Xregister struct   opnd *op1, *op2;
X{
X    if (op1->amode != op2->amode || op1->areg != op2->areg ||
```

```
X        op1->ireg  != op2->ireg)
X         return FALSE;
X
X    /*
X     * Depending on the addressing mode, we either need to
X     * compare the "astr" strings, or the displacements.
X     */
X    if ((op1->amode == ABS) || (op1->amode == (IMM|SYMB))) {
X         /* compare strings */
X         if (op1->astr == NULL)
X              return (op2->astr == NULL);
X         else {
X              if (op2->astr == NULL)
X                   return FALSE;
X
X              return (strcmp(op1->astr, op2->astr) == 0);
X         }
X    } else {
X         /* compare displacements */
X         return (op1->disp == op2->disp);
X    }
X}
END_OF_FILE
if test 7882 -ne `wc -c <'top/INST.C'`; then
    echo shar: \"'top/INST.C'\" unpacked with wrong size!
fi
# end of 'top/INST.C'
fi
if test -f 'top/PEEP1.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/PEEP1.C'\"
else
echo shar: Extracting \"'top/PEEP1.C'\" \(5178 characters\)
sed "s/^X//" >'top/PEEP1.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X
X/*
X * Single-instruction peephole optimizations and the overall driver routine.
X */
X
X#include "top.h"
X
X
Xvoid
Xpeep(bp)
```

```
Xregister BLOCK    *bp;
X{
X    bool peep1(), peep2(), peep3();
X    extern    BLOCK     *fhead;
X    register bool changed;
X
X    peep1(bp);
X
X    /*
X     * Loop until no more changes are made. After each change, do
X     * live/dead analysis or the data gets old. In each loop, make
X     * at most one change.
X     */
X    do {
X         changed = peep3(bp);
X
X         if (!changed)
X              changed = peep2(bp);
X
X         if (!changed)
X              changed = peep1(bp);
X
X         rhealth(fhead, FALSE);
X
X    } while (changed);
X}
X
X/*
X * ipeep1(ip) - check for changes to the instruction 'ip'
X */
Xstatic  bool
Xipeep1(bp, ip)
Xregister BLOCK    *bp;
Xregister INST     *ip;
X{
X    /*
X     * clr.l  Dn           => moveq.l  Dn
X     */
X    if (ip->opcode == CLR && ip->src.amode == REG &&
X        ISD(ip->src.areg) && (ip->flags & LENL)) {
X         ip->opcode = MOVEQ;
X         ip->dst = ip->src; /* we'll have two operands now */
X         ip->src.amode = IMM;
X         ip->src.disp = 0;
X         DBG(printf("%d ", __LINE__))
X         return TRUE;
X    }
X
X    /*
X     * move.*  #n,Dn       => moveq.l  #n,Dn
X     *
X     * moveq is always a long operation, but as long as the immediate
X     * value is appropriate, we don't care what the original length
```

```
X     * was. Clearing upper bytes won't matter.
X     */
X    if (ip->opcode == MOVE && ip->src.amode == IMM && ISD(ip->dst.areg) &&
X        D8OK(ip->src.disp)) {
X         ip->opcode = MOVEQ;
X         ip->flags = LENL;
X         DBG(printf("%d ", __LINE__))
X         return TRUE;
X    }
X
X    /*
X     * add.x  #n, X         => addq.x  #n, X
X     *
X     * where 1 <= n <= 8
X     */
X    if (ip->opcode == ADD && ip->src.amode == IMM &&
X        ip->src.disp >= 1 && ip->src.disp <= 8) {
X         ip->opcode = ADDQ;
X         DBG(printf("%d ", __LINE__))
X         return TRUE;
X    }
X
X    /*
X     * sub.x  #n, X         => subq.x  #n, X
X     *
X     * where 1 <= n <= 8
X     */
X    if (ip->opcode == SUB && ip->src.amode == IMM &&
X        ip->src.disp >= 1 && ip->src.disp <= 8) {
X         ip->opcode = SUBQ;
X         DBG(printf("%d ", __LINE__))
X         return TRUE;
X    }
X
X    /*
X     * movem.x  Reg,-(sp)       => move.x Reg,-(sp)
X     */
X    if (ip->opcode == MOVEM && ip->src.amode == REG &&
X        ip->dst.areg == SP && ip->dst.amode == (REGI|DEC)) {
X         ip->opcode = MOVE;
X         DBG(printf("%d ", __LINE__))
X         return TRUE;
X    }
X
X    /*
X     * movem.x  (sp)+,Reg       => move.x (sp)+,Reg
X     */
X    if (ip->opcode == MOVEM && ip->dst.amode == REG &&
X        ip->src.amode == (REGI|INC) && ip->src.areg == SP) {
X         ip->opcode = MOVE;
X         DBG(printf("%d ", __LINE__))
X         return TRUE;
X    }
```

```
X
X    /*
X     *   add[q]    #?, Rn
X     *
X     * Remove instruction if Rn is dead. This is most often used
X     * to eliminate the fixup of SP following a function call when
X     * we're just about to return, since the "unlk" clobbers SP
X     * anyway.
X     */
X    if ((ip->opcode == ADDQ || ip->opcode == ADD) && ip->src.amode == IMM &&
X         ip->dst.amode == REG) {
X
X         if ((ip->live & RM(ip->dst.areg)) == 0) {
X                   delinst(bp, ip);
X              DBG(printf("%d ", __LINE__))
X                   return TRUE;
X         }
X    }
X
X    /*
X     *   move.x    X, X
X     *
X     *   Delete as long as X isn't INC or DEC
X     */
X    if ((ip->opcode == MOVE) && opeq(&ip->src, &ip->dst) &&
X        ((ip->src.amode & (INC|DEC)) == 0)) {
X
X         delinst(bp, ip);
X         DBG(printf("%d ", __LINE__))
X         return TRUE;
X    }
X
X    /*
X     *   move.x    Rm, Rn
X     *
X     *   Delete if Rn is dead.
X     */
X    if (ip->opcode == MOVE &&
X         ip->src.amode == REG && ip->dst.amode == REG) {
X
X         if ((ip->live & RM(ip->dst.areg)) == 0) {
X                   delinst(bp, ip);
X              DBG(printf("%d ", __LINE__))
X                   return TRUE;
X         }
X    }
X
X
X    /*
X     *   cmp.x     #0, X          =>   tst.x     X
X     *   beq/bne                  beq/bne
X     *
X     *   Where X is not An
```

```
X     */
X    if (bp->last == ip && (bp->bcode == BEQ || bp->bcode == BNE) &&
X        ip->opcode == CMP &&
X        ((ip->dst.amode != REG) || !ISA(ip->dst.areg))) {
X
X         if (ip->src.amode == IMM && ip->src.disp == 0) {
X              ip->opcode = TST;
X              ip->src = ip->dst;
X              ip->dst.amode = NONE;
X              DBG(printf("%d ", __LINE__))
X              return TRUE;
X         }
X    }
X
X    /*
X     * add.x  #n, Am       => lea  n(Am), Am
X     *
X     * where 'n' is a valid displacement
X     */
X    if (ip->opcode == ADD && ip->src.amode == IMM && ip->dst.amode == REG &&
X        ISA(ip->dst.areg) && DOK(ip->src.disp)) {
X         ip->opcode = LEA;
X         ip->flags = 0;
X         ip->src.amode = REGID;
X         ip->src.areg = ip->dst.areg;
X         DBG(printf("%d ", __LINE__))
X         return TRUE;
X    }
X
X    return FALSE;
X}
X
X/*
X * peep1(bp) - peephole optimizations with a window size of 1
X */
Xstatic  bool
Xpeep1(bp)
Xregister BLOCK    *bp;
X{
X    register INST *ip;
X    register bool changed = FALSE;
X    register bool bchange;
X
X    DBG(printf("p1: "))
X    for (; bp != NULL ;bp = bp->next) {
X         bchange = FALSE;
X         for (ip = bp->first; ip != NULL ;ip = ip->next) {
X              if (ipeep1(bp, ip)) {
X                   s_peep1++;
X                   changed = TRUE;
X                   bchange = TRUE;
X              }
X         }
```

```
X         if (bchange)
X              bprep(bp);
X    }
X    DBG(printf("\n"); fflush(stdout))
X    return changed;
X}
END_OF_FILE
if test 5178 -ne `wc -c <'top/PEEP1.C'`; then
    echo shar: \"'top/PEEP1.C'\" unpacked with wrong size!
fi
# end of 'top/PEEP1.C'
fi
echo shar: End of archive 2 \(of 9\).
cp /dev/null ark2isdone
MISSING=""
for I in 1 2 3 4 5 6 7 8 9 ; do
    if test ! -f ark${I}isdone ; then
     MISSING="${MISSING} ${I}"
    fi
done
if test "${MISSING}" = "" ; then
    echo You have unpacked all 9 archives.
    rm -f ark[1-9]isdone ark[1-9][0-9]isdone
else
    echo You still need to unpack the following archives:
    echo "        " ${MISSING}
fi
##  End of shell archive.
exit 0
```

```
From 8346199399288417074
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,1a0a119ce9271ac7,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-24 18:47:44 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!uwm.edu!gem.mps.ohio-state.edu!
apple!sun-barr!newstop!sun!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i094: sozobon1.2 -- Update to Sozobon C compiler  part03/09
Keywords: shar, C
Message-ID: <34605@grapevine.uucp>
Date: 25 Oct 89 01:47:44 GMT
Sender: news@grapevine.EBay
Reply-To: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Lines: 1988
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Posting-number: Volume 2, Issue 94
Archive-name: sozobon1.2/part03
```

```
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of archive 3 (of 9)."
# Contents:  hcc/FIX.C hcc/GSUB.C hcc/NODES.C top/DATA.C
# Wrapped by koreth@panarthea on Tue Oct 24 18:40:45 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'hcc/FIX.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/FIX.C'\"
else
echo shar: Extracting \"'hcc/FIX.C'\" \(9053 characters\)
sed "s/^X//" >'hcc/FIX.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X * 
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  fix.c
X *
X *  Motorola has is not consistent in what operations allow
X *  which operands.  This section compensates for that --
X *  tries to find the best way to do something.
X */
X
X#include <stdio.h>
X#include "param.h"
X#include "nodes.h"
X#include "flags.h"
X#include "bstok.h"
X#include "gen.h"
X#include "ops.h"
X
X#define FAIL 0
X#define OKAY 1
X
X#define CL_IMM    IOPD
X#define CL_AREG   AOPD
X#define CL_DREG DOPD
X#define CL_ADR    MOPD
X
X#define RETAREG   0x100
X#define RETDREG 0x200
X#define RETLEFT   0x400
X#define RETRIGHT 0x800
X
```

```
X#define LISTMP    1
X#define RISTMP    2
X#define NDASSOC   4
X#define CANRL     8
X#define CANLR     0x10
X#define CANRD     0x20
X#define CANDL     0x40
X#define CANLD     0x80
X#define CANDD     0x100
X#define CANDR     0x200
X
Xstruct mtbl {
X    int restr;
X    char *code;
X    int needregs;
X} tbl2[] = {
X    {LISTMP|CANRL,
X         "\tN.S\t>A,<A\n", RETLEFT},
X    {RISTMP|CANLR|NDASSOC,
X         "\tN.S\t<A,>A\n", RETRIGHT},
X    {CANRD,
X         "\tmove.S\t<A,A\n\tN.S\t>A,A\n", RETDREG},
X    {CANLD|NDASSOC,
X         "\tmove.S\t>A,A\n\tN.S\t<A,A\n", RETDREG},
X    /* only EOR and shifts get to here */
X    {CANDL|LISTMP,
X         "\tmove.S\t>A,R1\n\tN.S\tR1,A\n", RETLEFT|1},
X    {CANDR|RISTMP|NDASSOC,
X         "\tmove.S\t<A,R1\n\tN.S\tR1,A\n", RETRIGHT|1},
X    {CANDD,
X         "\tmove.S\t<A,A\n\tmove.S\t>A,R1\n\tN.S\tR1,A\n",
X             RETDREG|1},
X    {0}
X};
X
Xstruct mtbl tblmul[] = {
X    {CANRL|LISTMP,
X         "\tmulU\t>A,<A\n", RETLEFT},
X    {CANLR|RISTMP,
X         "\tmulU\t<A,>A\n", RETRIGHT},
X    {CANRD,
X         "\tmove.w\t>A,A\n\tmulU\t<A,A\n", RETDREG},
X    {0}
X};
X
Xstruct mtbl tbldiv[] = {
X    {CANRL|LISTMP,
X         "\text.l\t<A\n\tdivs\t>A,<A\n", RETLEFT},
X    {CANRD,
X         "\tmove.w\t<A,A\n\text.l\tA\n\tdivs\t>A,A\n", RETDREG},
X    {0}
X};
X
```

```
Xstruct mtbl tbludiv[] = {
X    {CANRD,
X         "\tclr.l\tA\n\tmove.w\t<A,A\n\tdivu\t>A,A\n", RETDREG},
X    {0}
X};
X
Xstruct mtbl tblc[] = {
X    {CANRL,
X         "\tcmp.<S\t>A,<A\n", 0},
X    {CANRD,
X         "\tmove.<S\t<A,R1\n\tcmp.<S\t>A,R1\n", 1},
X/* shouldnt get to here! */
X    {CANDL,
X         "\tmove.<S\t>A,R1\n\tcmp.<S\tR1,<A\n", 1},
X    {CANDD,
X         "\tmove.<S\t>A,R1\n\tmove.<S\t<A,R2\n\tcmp.<S\tR1,R2\n", 2},
X    {0}
X};
X
Xstruct mtbl tblas[] = {
X    {CANRL,
X         "\tN.S\t>A,<A\n", RETLEFT},
X    {CANDL,
X         "\tmove.S\t>A,R1\n\tN.S\tR1,A\n", 1|RETLEFT},
X    /* only MUL, DIV and shifts should get to here */
X    {CANRD,
X         "\tmove.S\t<A,A\n\tN.S\t>A,A\n\tmove.S\tA,<A\n",
X             RETDREG},
X    {CANLD|NDASSOC,
X         "\tmove.S\t>A,A\n\tN.S\t<A,A\n\tmove.S\tA,<A\n",
X             RETDREG},
X    {CANDD,
X    "\tmove.S\t<A,A\n\tmove.S\t>A,R1\n\tN.S\tR1,A\n\tmove.S\tA,<A\n",
X             RETDREG|1},
X    {0}
X};
X
Xstruct mtbl tblamul[] = {
X    {CANRL,
X         "\tmulU\t>A,<A\n", RETLEFT},
X    {CANLR|RISTMP,
X         "\tmulU\t<A,>A\n\tmove.w\t>A,<A\n", RETRIGHT},
X    {CANRD,
X         "\tmove.w\t<A,A\n\tmulU\t>A,A\n\tmove.w\tA,<A\n", RETDREG},
X    {0}
X};
X
Xstruct mtbl tbladiv[] = {
X    {CANRL,
X         "\text.l\t<A\n\tdivs\t>A,<A\n", RETLEFT},
X    {CANRD,
X    "\tmove.w\t<A,A\n\text.l\tA\n\tdivs\t>A,A\n\tmove.w\tA,<A\n",
X         RETDREG},
```

```
X    {0}
X};
X
Xstruct mtbl tbluadiv[] = {
X    {CANRD,
X    "\tclr.l\tA\n\tmove.w\t<A,A\n\tdivu\t>A,A\n\tmove.w\tA,<A\n",
X         RETDREG},
X    {0}
X};
X
Xstruct mtbl tblamod[] = {
X    {CANRL,
X         "\text.l\t<A\n\tdivs\t>A,<A\n\tswap\t<A\n", RETLEFT},
X    {CANRD,
X  "\tmove.w\t<A,A\n\text.l\tA\n\tdivs\t>A,A\n\tswap\tA\n\tmove.w\tA,<A\n",
X         RETDREG},
X    {0}
X};
X
Xstruct mtbl tbluamod[] = {
X    {CANRD,
X  "\tclr.l\tA\n\tmove.w\t<A,A\n\tdivu\t>A,A\n\tswap\tA\n\tmove.w\tA,<A\n",
X         RETDREG},
X    {0}
X};
X
Xclass(np)
XNODEP np;
X{
X    switch (np->g_token) {
X    case ICON:
X         return CL_IMM;
X    case ONAME:
X         return (np->g_flags & IMMEDID) ? CL_IMM : CL_ADR;
X    case OREG:
X         return (np->g_flags & IMMEDID) ? CL_IMM : CL_ADR;
X    case PUSHER:
X         return CL_ADR;
X    case REGVAR:
X         if (np->g_rno < AREG)
X              return CL_DREG;
X         else
X              return CL_AREG;
X    default:
X         printf("Weird class ");
X         return CL_IMM;
X    }
X}
X
Xint canswap;
XNODEP matchnp;
X
Xfix_cmp(np, flags)
```

```
XNODEP np;
X{
X    NODEP rp = np->n_right;
X
X    /* special stuff for ICON 0 */
X    if (rp->g_token == ICON && rp->g_offs == 0 &&
X         (rp->g_flags & IMMEDID)) {
X         addcode(np, "<Q");
X         return OKAY;
X    }
X    canswap = 0;
X    return fix_sub(np, flags, tblc);
X}
X
Xfix2ops(np, flags)
XNODEP np;
X{
X    canswap = flags & ASSOC;
X    return fix_sub(np, flags, tbl2);
X}
X
Xfixmul(np, flags)
XNODEP np;
X{
X    return fix_sub(np, flags, tblmul);
X}
X
Xfixdiv(np, flags)
XNODEP np;
X{
X    return fix_sub(np, flags, np->g_ty == ET_U ? tbludiv : tbldiv);
X}
X
Xfixmod(np, flags)
XNODEP np;
X{
X    int r;
X
X    r = fixdiv(np, flags);
X    addcode(np, "\tswap\tA\n");
X    return r;
X}
X
Xfixamul(np, flags)
XNODEP np;
X{
X    return fix_sub(np, flags, tblamul);
X}
X
Xfixadiv(np, flags)
XNODEP np;
X{
X    return fix_sub(np, flags, np->g_ty == ET_U ? tbluadiv : tbladiv);
```

```
X}
X
Xfixamod(np, flags)
XNODEP np;
X{
X    return fix_sub(np, flags, np->g_ty == ET_U ? tbluamod : tblamod);
X}
X
Xfix_asn(np, flags)
XNODEP np;
X{
X    canswap = 0;
X    return fix_sub(np, flags, tblas);
X}
X
Xfix_sub(np, flags, tbl)
Xregister NODEP np;
Xstruct mtbl *tbl;
X{
X    NODEP lp = np->n_left, rp = np->n_right;
X    int lclass, rclass;
X    register struct mtbl *p;
X    int rv;
X
X    lclass = class(lp);
X    rclass = class(rp);
X    matchnp = np;
X
X    for (p = tbl; p->restr; p++)
X         if ((rv = cando(flags, p->restr, lclass, rclass))) {
X              dotbl(p, np);
X              return OKAY;
X         }
X    printf("no code table match! ");
X    return FAIL;
X}
X
Xdotbl(p, np)
Xstruct mtbl *p;
XNODEP np;
X{
X    register int i, k;
X
X    if (p->needregs) {
X         i = p->needregs;
X         if (i & RETLEFT)
X              inherit(np);
X         else if (i & RETRIGHT)
X              rinherit(np);
X         else if (i & RETAREG)
X              retreg(np, ralloc(AREG));
X         else if (i & RETDREG)
X              retreg(np, ralloc(0));
```

```
X         k = i & 7;
X         while (k--)
X              tempr(np, 0);
X         k = (i/AREG) & 7;
X         while (k--)
X              tempr(np, AREG);
X    }
X    addcode(np, p->code);
X}
X
X#define canflag(l,r)  (l << cansh[r])
X
Xshort cansh[] = {0, 0, 4, 0, 8, 0, 0, 0, 12};
X
Xcando(flags, restr, lc, rc)
Xregister restr;
X{
X    NODEP lp = matchnp->n_left, rp = matchnp->n_right;
X
X    if (restr & RISTMP) {
X         if ((rc & (CL_DREG|CL_AREG)) == 0 ||
X              istemp(rp->g_rno) == 0)
X              return 0;
X    } else if (restr & LISTMP) {
X         if ((lc & (CL_DREG|CL_AREG)) == 0 ||
X              istemp(lp->g_rno) == 0)
X              return 0;
X    }
X    if (restr & NDASSOC) {
X         if (canswap == 0)
X              return 0;
X    }
X    if (restr & (CANLR|CANLD)) {
X         flags |= quickflag(lp, flags);
X    } else if (restr & (CANRL|CANRD)) {
X         flags |= quickflag(rp, flags);
X    }
X    flags &= 0xfff;
X    if (restr & CANLR) {
X         if ((flags & canflag(lc,rc)) == 0)
X              return 0;
X    } else if (restr & CANRL) {
X         if ((flags & canflag(rc,lc)) == 0)
X              return 0;
X    } else if (restr & CANRD) {
X         if ((flags & canflag(rc,CL_DREG)) == 0)
X              return 0;
X    } else if (restr & CANDL) {
X         if ((flags & canflag(CL_DREG,lc)) == 0)
X              return 0;
X    } else if (restr & CANLD) {
X         if ((flags & canflag(lc,CL_DREG)) == 0)
X              return 0;
```

```
X    } else if (restr & CANDR) {
X         if ((flags & canflag(CL_DREG,rc)) == 0)
X              return 0;
X    } else if (restr & CANDD) {
X         if ((flags & DOPD) == 0)
X              return 0;
X    }
X    return 1;
X}
X
Xquickflag(np, flags)
XNODEP np;
X{
X    long offs;
X
X    if (np->g_token != ICON)
X         return 0;
X    offs = np->g_offs;
X    if ((flags & QOPD) && np->g_offs >= 1 && np->g_offs <= 8)
X         return IOPD;
X    if ((flags & ONEOPM) && np->g_offs == 1)
X         return IOPM;
X    return 0;
X}
X
X/*
X * stuff to do field extract and field assign
X * Yes, this stuff can be optimized a lot more ... feel free
X */
Xfldget(np, flags)
Xregister NODEP np;
X{
X    int how;
X
X    how = 1;
X    if (np->g_fldo == 0)
X         how = 0;  /* no shift needed */
X    else if (np->g_fldo > 8)
X         how = 2;  /* shift too big for immediate */
X
X    np->n_left->g_offs += np->g_offs;     /* major HACK, hope it works */
X
X    retreg(np, ralloc(0));
X
X    addcode(np, "\tmove\t<A,A\n");        /* get word */
X    if (how)                /* shift to position */
X         addcode(np, how==1 ?    "\tlsr\t#Y,A\n" :
X              "\tlsr\t#8,A\n\tlsr\t#Z,A\n");
X    addcode(np, "\tand\t#W,A\n");         /* mask off top */
X
X    return OKAY;
X}
X
```

```
Xfldasn(np, flags)
Xregister NODEP np;
X{
X    int how;
X
X    how = 1;
X    if (np->g_fldo == 0)
X         how = 0; /* no shift needed */
X    else if (np->g_fldo > 8)
X         how = 2; /* shift too big for immediate */
X
X    retreg(np, ralloc(0));
X
X    tempr(np, 0);
X
X    addcode(np, "\tmove\t>A,A\n");       /* value */
X    if (needmask(np))      /* mask off high bits */
X         addcode(np, "\tand\t#W,A\n");
X    if (how) /* shift into position */
X         addcode(np, how==1 ?   "\tlsl\t#Y,A\n" :
X                        "\tlsl\t#8,A\n\tlsl\t#Z,A\n");
X
X    addcode(np, "\tmove\t<A,R1\n");      /* old value */
X    addcode(np, "\tand\t#X,R1\n");       /* mask out old value */
X    addcode(np, "\tor\tA,R1\n");         /* set new field */
X    addcode(np, "\tmove\tR1,<A\n");      /* store new word */
X
X    if (how && (flags & NOVAL_OK) == 0)  /* return value */
X         addcode(np, how==1 ?   "\tlsr\t#Y,A\n" :
X                        "\tlsr\t#8,A\n\tlsr\t#Z,A\n");
X
X    return OKAY;
X}
X
Xneedmask(np)
Xregister NODEP np;
X{
X    NODEP lp = np->n_left;
X
X    if (np->g_fldw + np->g_fldo >= 16)
X         return 0;
X
X    if (lp->g_token != ICON)
X         return 1;
X    if ((int)lp->g_offs & ~ones(np->g_fldw)) {
X         warn("value too big for field");
X         return 1;
X    }
X    return 0;
X}
END_OF_FILE
if test 9053 -ne `wc -c <'hcc/FIX.C'`; then
    echo shar: \"'hcc/FIX.C'\" unpacked with wrong size!
```

```
fi
# end of 'hcc/FIX.C'
fi
if test -f 'hcc/GSUB.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/GSUB.C'\"
else
echo shar: Extracting \"'hcc/GSUB.C'\" \(9472 characters\)
sed "s/^X//" >'hcc/GSUB.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X * 
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  gsub.c
X *
X *  Various code generation subroutines
X *  Includes generation of switches and
X *  conversion of type lists to simple type,size.
X */
X
X#include <stdio.h>
X#include "param.h"
X#include "bstok.h"
X#include "tytok.h"
X#include "flags.h"
X#include "nodes.h"
X#include "gen.h"
X
X#define isimmed(np)    ((np)->g_flags & IMMEDID)
X#define isareg(np)     ((np)->g_token == REGVAR && (np)->g_rno >= AREG)
X
Xextern cctest;
X
Xextern xflags[];
X#define debug     xflags['s'-'a']
X
Xgen_brt(np, lbl)
XNODEP np;
X{
X    p2_expr(&np);
X    mustty(np, R_SCALAR);
X    br_sub(np, 0, lbl);
X}
X
Xgen_brf(np, lbl)
XNODEP np;
X{
X    p2_expr(&np);
```

```
X    mustty(np, R_SCALAR);
X    br_sub(np, 1, lbl);
X}
X
Xbr_sub(np, rev, lbl)
XNODEP np;
X{
X    int i;
X
X    switch (np->e_token) {
X         case DOUBLE '&':
X              br_split(np, lbl, rev);
X              return;
X         case DOUBLE '|':
X              br_split(np, lbl, 2+rev);
X              return;
X    }
X    genx(np, FORCC);
X    i = cctest;
X    if (i) {
X         if (rev)
X              /* reverse truth */
X              i = (i&1) ? i+1 : i-1;
X         out_b(i, lbl);
X    }
X}
X
Xbr_split(np, lbl, n)
XNODEP np;
X{
X    int nlbl;
X
X    if (n == 0 || n == 3)
X         nlbl = new_lbl();
X    else
X         nlbl = lbl;
X
X    br_sub(np->n_left, n<2, nlbl);
X    br_sub(np->n_right, n&1, lbl);
X
X    freeunit(np);
X
X    if (nlbl != lbl)
X         def_lbl(nlbl);
X}
X
X/* generate switch
X    np - list of nodes with value,label pairs (sorted)
X    dlbl - default label or -1
X */
X#undef min
X#undef max
X
```

```
Xgen_switch(np, odlbl)
XNODEP np;
X{
X    int n,min,max;
X    int dlbl;
X    register NODEP p;
X
X    if (debug) {
X         printf("gs %d ", odlbl);
X         printnode(np);
X    }
X
X    /* if no default, make one! */
X    if (odlbl < 0)
X         dlbl = new_lbl();
X    else
X         dlbl = odlbl;
X
X    n = 0;
X    for (p=np; p; p=p->n_next) {
X         if (n == 0)
X              min = max = p->c_casev;
X         else
X              max = p->c_casev;
X         n++;
X    }
X    if (n <= C_SIMPLE)
X         simple_sw(np,odlbl);
X    else if (n >= max/C_RATIO - min/C_RATIO)
X         table_sw(np,dlbl,min,max);
X    else {
X         half_sw(np,dlbl,max/2+min/2,n);
X         goto out;     /* free already done */
X    }
X    freenode(np);
Xout:
X    if (odlbl < 0)
X         def_lbl(dlbl);
X}
X
X/* simple if-else type switch
X    dlbl may be -1 -> fall through
X    does not free np
X*/
Xsimple_sw(np, dlbl)
Xregister NODEP np;
X{
X    while (np) {
X         out_d0cmp(np->c_casev);
X         out_b(B_EQ, np->c_casel);
X         np = np->n_next;
X    }
X    if (dlbl >= 0)
```

```
X        out_br(dlbl);
X}
X
X/* use table switch
X   dlbl is not -1
X   does not free np
X*/
Xtable_sw(np, dlbl, min, max)
XNODEP np;
X{
X   out_d0cmp(min);
X   out_b(B_LT, dlbl);
X   out_d0cmp(max);
X   out_b(B_GT, dlbl);
X   if (min)
X        out_d0sub(min);
X
X   out_tsw();
X
X   while (np) {
X        while (min < np->c_casev) {
X            out_tlbl(dlbl);
X            min++;
X        }
X        out_tlbl(np->c_casel);
X        min++;
X        np = np->n_next;
X   }
X}
X
X/* cut switch in half (by value)
X   dlbl is not -1
X   will free np
X */
Xhalf_sw(np, dlbl, cut, n)
XNODEP np;
X{
X   register NODEP p, last;
X   int nlo, nhi;
X   int l1;
X
X   for (p=np; p->c_casev < cut; p = p->n_next)
X        last = p;
X   /* we KNOW both pieces are non-NULL ! */
X   last->n_next = NULL;
X   last = p;
X
X   nlo = 0;
X   nhi = 0;
X   for (p=np; p; p=p->n_next)
X        nlo++;
X   nhi = n - nlo;
X
```

```
X    if (nhi == 1) {        /* switch hi and low */
X         p = np;
X         np = last;
X         last = p;
X         nlo = 1;
X         nhi = n-1;
X    }
X    if (nlo == 1) {        /* also nhi == 1 */
X         out_d0cmp(np->c_casev);
X         out_b(B_EQ, np->c_casel);
X         freenode(np);
X         gen_switch(last, dlbl);
X         return;
X    }
X    l1 = new_lbl();
X    out_d0cmp(cut);
X    out_b(B_GE, l1);
X    gen_switch(np, dlbl);
X    def_lbl(l1);
X    gen_switch(last, dlbl);
X}
X
Xistempa(np)
Xregister NODEP np;
X{
X    if (np->g_token == OREG && istemp(np->g_rno))
X         return 1;
X    return 0;
X}
X
Xchar chunkch[] = {0, 'b', 'w', 0, 'l'};
X
Xstrasn(np)
XNODEP np;
X{
X    int r;
X    long count, i;
X    int chunk, l;
X    char buf[40];
X    int lisa, risa, rissp;
X    int useloop;
X
X    if (np->g_ty != ET_A)
X         return 0;
X
X    count = np->g_bsize;
X    chunk = np->g_sz;
X    if (count == 1) {
X         return 0;
X    }
X
X    useloop = count > 3;
X
```

```
X    lisa = istempa(np->n_left);
X    risa = istempa(np->n_right);
X    rissp = (np->n_right->g_token == PUSHER);
X
X    if (lisa)
X         r = np->n_left->g_rno;
X    else
X         r = ralloc(AREG);  /* R0 */
X    indir(np, r);
X    np->g_offs = -(count*chunk);
X
X    if (risa)
X         np->g_r1 = np->n_right->g_rno;
X    else if (rissp)
X         np->g_r1 = AREG+7;
X    else
X         tempr(np, AREG);   /* R1 */
X    if (useloop)
X         tempr(np, 0);      /* R2 */
X
X    if (!lisa || np->n_left->g_offs)
X         addcode(np, "\tlea\t<A,R0\n");
X    if (!(risa||rissp) || np->n_right->g_offs)
X         addcode(np, "\tlea\t>A,R1\n");
X
X    if (useloop) {
X         sprintf(buf, "\tmove.w\t#%d,R2\n", (int)(count-1));
X         addcode(np, buf);
X         l = new_lbl();
X         sprintf(buf, "'L%d:\tmove.%c\t(R1)+,(R0)+\n",l,chunkch[chunk]);
X         addcode(np, buf);
X         sprintf(buf, "\tdbra\tR2,'L%d\n", l);
X         addcode(np, buf);
X    } else for (i=0; i<count; i++) {
X         sprintf(buf, "\tmove.%c\t(R1)+,(R0)+\n", chunkch[chunk]);
X         addcode(np, buf);
X    }
X
X    if (chunk == 1 && count & 1 && rissp)
X         addcode(np, "\tmove.b\t(sp)+,R2\n");
X
X    return 1;
X}
X
Xstrret(np)
XNODEP np;
X{
X    extern funstrl;
X
X    strsub(np, funstrl);
X}
X
Xstrpush(np)
```

```
XNODEP np;
X{
X    return strsub(np, 0);
X}
X
Xstrxpush(np)
XNODEP np;
X{
X    strsub(np, -1);
X}
X
Xstrsub(np, tolbl)
Xregister NODEP np;
X{
X    long count, i;
X    int chunk, l;
X    char buf[40];
X    char *frstr;
X    register NODEP tp;
X    int useloop;
X
X    count = np->g_bsize;
X    chunk = np->g_sz;
X    if (chunk == 1 && count & 1 && tolbl <= 0)
X         count++;
X
X    useloop = count > 3;
X
X    if (useloop && np->g_r2 == -1)
X         np->g_r2 = ralloc(0);
X
X    /* set up 'from' address */
X    if (np->g_token == OREG && istemp(np->g_rno)) {
X         frstr = "R0";
X         if (np->g_offs)
X              addcode(np, "\tlea\tA,R0\n");
X    } else if (tolbl == -1 && (tp = np->n_left) &&
X        tp->g_token == OREG && istemp(tp->g_rno) &&
X        tp->g_offs == 0) {
X         frstr = "R1";
X         np->g_r1 = tp->g_rno;
X    } else {
X         frstr = "R1";
X         if (np->g_r1 == -1)
X              np->g_r1 = ralloc(AREG);
X         if (tolbl == -1)
X              addcode(np, "\tlea\t<A,R1\n");
X         else
X              addcode(np, "\tlea\tA,R1\n");
X    }
X
X    /* set up 'to' address */
X    if (tolbl > 0) {
```

```
X         sprintf(buf, "\tmove.l\t#'L%d,H\n", tolbl);
X         addcode(np, buf);
X    } else {
X         sprintf(buf, "\tsub\t#%d,sp\n", (int)(count*chunk));
X         addcode(np, buf);
X         addcode(np, "\tmove.l\tsp,H\n");
X    }
X
X    /* generate copy loop */
X    if (useloop) {
X         sprintf(buf, "\tmove.w\t#%d,R2\n", (int)(count-1));
X         addcode(np, buf);
X         l = new_lbl();
X         sprintf(buf, "'L%d:\tmove.%c\t(%s)+,(H)+\n", l, chunkch[chunk],
X              frstr);
X         addcode(np, buf);
X         sprintf(buf, "\tdbra\tR2,'L%d\n", l);
X         addcode(np, buf);
X    } else for (i=0; i<count; i++) {
X         sprintf(buf, "\tmove.%c\t(%s)+,(H)+\n", chunkch[chunk],
X              frstr);
X         addcode(np, buf);
X    }
X
X    return count*chunk;
X}
X
Xspecasn(np, flags)
XNODEP np;
X{
X    NODEP lp = np->n_left, rp = np->n_right;
X    int r;
X
X    if (rp->g_token == ICON && isimmed(rp)) {
X         rinherit(np);
X
X         if (rp->g_offs == 0)
X              addcode(np, isareg(lp) ? "\tsub.S\t<A,<A\n"
X                   : "\tclr.S\t<A\n");
X         else
X              addcode(np, "\tmove.S\t>A,<A\n");
X         return 1;
X    }
X    if (rp->g_token == OREG && isimmed(rp)) {
X         rp->g_flags &= ~IMMEDID;
X         if (isareg(lp)) {
X              inherit(np);
X              addcode(np, "\tlea\t>A,A\n");
X         } else if (rp->g_offs == 0) {
X              retreg(np, rp->g_rno);
X              addcode(np, "\tmove.l\tA,<A\n");
X         } else {
X              if (istemp(rp->g_rno))
```

```
X                 r = rp->g_rno;
X             else
X                 r = ralloc(AREG);
X             retreg(np, r);
X             addcode(np, "\tlea\t>A,A\n");
X             addcode(np, "\tmove.l\tA,<A\n");
X         }
X         return 1;
X    }
X    return 0;
X}
X
Xuntype(np)
Xregister NODEP np;
X{
X    if (np == NULL || np->n_tptr == NULL) {
X         printf("? NULL untype ");
X         return;
X    }
X
X    switch (np->e_type) {
X    case E_BIN:
X         untype(np->n_right);
X         /* fall through */
X    case E_UNARY:
X         if (np->e_token == '.' && np->e_fldw) {
X             np->g_fldw = np->e_fldw;
X             np->g_fldo = np->e_fldo;
X         } else
X             np->g_fldw = 0;
X
X         untype(np->n_left);
X    }
X
X    get_tyinf(np);
X
X    if ((np->n_flags & N_COPYT) == 0)
X         freenode(np->n_tptr);
X    np->n_tptr = NULL;      /* is g_code */
X    np->g_betw = NULL;
X}
X
Xstatic char bty[] = {
X    ET_U, ET_U, ET_S, ET_S, ET_U, ET_S, ET_S, ET_F, ET_F, 0
X};
X
Xstatic char bsz[] = {
X    SIZE_C, SIZE_L, SIZE_L, SIZE_S, SIZE_U,
X    SIZE_I, SIZE_C, SIZE_F, SIZE_D, 0
X};
X
Xget_tyinf(np)
Xregister NODEP np;
```

```
X{
X    NODEP tp = np->n_tptr;
X    int n;
X    long offs;
X
X    offs = np->e_offs;
X
X    /* inherit name,token,left,right,nmx from common
X         and token, flags, type, sc from enode */
X
X    switch (tp->t_token) {
X    case K_STRUCT:
X    case K_UNION:
X         np->g_ty = ET_A;
X         /* determine copy chunk size and number of chunks */
X         if (tp->t_aln & 1) {
X              if (tp->t_size & 1)
X                   fatal("aligned struct with odd size");
X              if (tp->t_size & 2) {
X                   np->g_bsize = tp->t_size >> 1;
X                   np->g_sz = 2;
X              } else {
X                   np->g_bsize = tp->t_size >> 2;
X                   np->g_sz = 4;
X              }
X         } else {
X              np->g_bsize = tp->t_size;
X              np->g_sz = 1;
X         }
X         break;
X    case '(':
X         break;
X    case STAR:
X         np->g_ty = ET_U;
X         np->g_sz = SIZE_P;
X         break;
X    default:
X         n = tp->t_token-FIRST_BAS;
X         np->g_ty = bty[n];
X         np->g_sz = bsz[n];
X    }
X    np->g_offs = offs; /* different place */
X}
X
Xaddcode(np, s)
Xregister NODEP np;
Xchar *s;
X{
X    NODEP tp;
X    int i, c;
X
X    while (np->g_code)
X         np = np->g_code;
```

```
X    tp = allocnode();
X    np->g_code = tp;
X    np->n_flags &= ~N_COPYT;
X    i = strlen(s);
X    if (i < NMXSIZE) { /* fits in one */
X         strcpy(tp->n_name, s);
X         return;
X    }
X
X    /* need to split it */
X    i = NMXSIZE-1;
X    c = s[i-1];
X    if (c == '<' || c == '>' || (c>='A' && c<='Z'))    /* special */
X         i--;
X    strncpy(tp->n_name, s, i);
X    tp->n_name[i] = 0;
X    addcode(tp, &s[i]);
X}
END_OF_FILE
if test 9472 -ne `wc -c <'hcc/GSUB.C'`; then
    echo shar: \"'hcc/GSUB.C'\" unpacked with wrong size!
fi
# end of 'hcc/GSUB.C'
fi
if test -f 'hcc/NODES.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/NODES.C'\"
else
echo shar: Extracting \"'hcc/NODES.C'\" \(8680 characters\)
sed "s/^X//" >'hcc/NODES.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  nodes.c
X *
X *  Node allocation, deallocation, searching, printing
X *  and other node handling
X */
X
X#include <stdio.h>
X#include "param.h"
X#include "nodes.h"
X
Xextern FILE *output;
XNODE *freelist;
X
X#define NODEINCR  100
```

```
X
Xextern int oflags[];
X#define debug oflags['n'-'a']
X
X#define NODELEN   (sizeof(NODE)/4)
X
Xint nodesmade, nodesavail;
X
XNODE *
Xallocnode()
X{
X    char *calloc();
X    NODE *t;
X    int i;
X
Xretry:
X    if (freelist != 0) {
X         t = freelist;
X         freelist = t->n_next;
X         lclr(t, NODELEN);
X         nodesavail--;
X         if (debug)
X              printf("%lx+ ", t);
X         return t;
X    }
X    t = (NODE *)calloc(NODEINCR, sizeof(NODE));
X    if (t == 0) {
X         printf("malloc failure\n");
X         exit(1);
X    }
X    nodesmade += NODEINCR;
X    nodesavail += NODEINCR;
X    for (i=0; i<NODEINCR; i++)
X         t[i].n_next = &t[i+1];
X    t[NODEINCR-1].n_next = 0;
X    freelist = t;
X    goto retry;
X}
X
Xfreeunit(t)
XNODE *t;
X{
X    if (t->n_flags & N_ISFREE) {
X         printf("%lx ", t);
X         error("Freeing free node");
X         exit(1);
X    } else
X         t->n_flags |= N_ISFREE;
X    t->n_next = freelist;
X    freelist = t;
X    nodesavail++;
X    if (debug)
X         printf("%lx- ", t);
```

```
X}
X
Xfreenode(t)
XNODE *t;
X{
X    register NODE *nxt;
X
X    if (t == NULL) return;
Xagain:
X    if (t->n_right)
X         freenode(t->n_right);
X    if (t->n_nmx)
X         freenode(t->n_nmx);
X    if (t->n_tptr && (t->n_flags & N_COPYT) == 0)
X         freenode(t->n_tptr);
X    nxt = t->n_left;
X    freeunit(t);
X    if (nxt) {
X         t = nxt;
X         goto again;   /* minimize left recursion */
X    }
X}
X
Xput_nnm(t)
XNODE *t;
X{
X    printf("%s", t->n_name);
X    while (t->n_nmx) {
X         t = t->n_nmx;
X         printf("%s", t->n_name);
X    }
X}
X
Xqput_nnm(t, fd)
XNODE *t;
XFILE *fd;
X{
X    fprintf(fd, "%s", t->n_name);
X    while (t->n_nmx) {
X         t = t->n_nmx;
X         fprintf(fd, "%s", t->n_name);
X    }
X}
X
Xfput_nnm(t)
XNODE *t;
X{
X    fprintf(output, "%s", t->n_name);
X    while (t->n_nmx) {
X         t = t->n_nmx;
X         fprintf(output, "%s", t->n_name);
X    }
X}
```

```
X
X/* add a short string (less than NMXSIZE) to front of name */
Xnnmins(t, s)
XNODEP t;
Xchar *s;
X{
X    register i, j;
X    char tbuf[NMSIZE];
X    NODEP n;
X
X    i = strlen(t->n_name);
X    j = strlen(s);
X    if (j > NMSIZE-1)
X         return;       /* compiler error */
X    if (i+j <= NMSIZE-1) {      /* fits in node */
X         strcpy(tbuf, t->n_name);
X         strcpy(t->n_name, s);
X         strcpy(t->n_name+j, tbuf);
X    } else {
X         n = allocnode();
X         n->n_nmx = t->n_nmx;
X         t->n_nmx = n;
X         strcpy(n->n_name, t->n_name);
X         strcpy(t->n_name, s);
X    }
X}
X
X/* add a short string (less than NMXSIZE) to end of name */
Xnnmadd(t, s)
XNODE *t;
Xchar *s;
X{
X    register i,j;
X    int sizeb;
X    NODEP n;
X
X    /* find last node */
X    sizeb = NMSIZE;
X    while (t->n_nmx) {
X         t = t->n_nmx;
X         sizeb = NMXSIZE;
X    }
X    /* fits in current last node? */
X    i = strlen(s);
X    j = strlen(t->n_name);
X    if (i < sizeb-j) {
X         strcat(t->n_name, s);
X         return;
X    }
X    /* put all of s in new node */
X    n = allocnode();
X    t->n_nmx = n;
X    t = n;
```

```
X    strncpy(t->n_name, s, NMXSIZE-1);
X    t->n_name[NMXSIZE-1] = 0;
X}
X
Xnscpy(t, s)
XNODE *t;
Xchar *s;
X{
X    register i;
X    NODEP n;
X
X    i = strlen(s);
X    strncpy(t->n_name, s, NMSIZE-1);
X    t->n_name[NMSIZE-1] = 0;
X    i -= NMSIZE-1;
X    s += NMSIZE-1;
X    while (i > 0) {
X         n = allocnode();
X         t->n_nmx = n;
X         t = n;
X         strncpy(t->n_name, s, NMXSIZE-1);
X         t->n_name[NMXSIZE-1] = 0;
X         i -= NMXSIZE-1;
X         s += NMXSIZE-1;
X    }
X}
X
Xputlist(head, np)
XNODE **head, *np;
X{
X    np->n_next = *head;
X    *head = np;
X}
X
Xputhlist(head, np)
XNODE *head[], *np;
X{
X    putlist(&head[hash(np->n_name)], np);
X}
X
XNODE *
Xllook(head, np)
XNODE *head, *np;
X{
X    register NODEP p;
X
X    for (p=head; p != NULL; p = p->n_next)
X         if (xstrcmp(p, np) == 0) {
X              return p;
X         }
X    return NULL;
X}
X
```

```
XNODE *
Xhlook(head, np)
XNODE *head[], *np;
X{
X    register NODEP p;
X
X    p = head[hash(np->n_name)];
X    return llook(p, np);
X}
X
Xhash(s)
Xregister char *s;
X{
X    register hval;
X
X    hval = 0;
X    while (*s)
X         hval += *s++;
X    return hval & (NHASH-1);
X}
X
Xxstrcmp(p1, p2)
XNODE *p1, *p2;
X{
X    int rv;
X
X    if ((rv = strcmp(p1->n_name, p2->n_name)) != 0)
X         return rv;
X    if (p1->n_nmx == NULL) {
X         if (p2->n_nmx == NULL)
X              return 0;
X         return -1;
X    }
X    if (p2->n_nmx == NULL)
X         return 1;
X    return xstrcmp(p1->n_nmx, p2->n_nmx);
X}
X
Xchar *
Xscopy(s)
Xchar *s;
X{
X    int i;
X    char *p;
X
X    i = strlen(s)+1;
X    if (i > sizeof(NODE)) {
X         error("preproc name too big");
X         i = sizeof(NODE);
X         s[i-1] = 0;
X    }
X    p = (char *)allocnode();
X    strcpy(p, s);
```

```
X    return p;
X}
X
Xsfree(s)
Xchar *s;
X{
X    NODEP np;
X
X    np = (NODEP)s;
X    np->n_flags = 0;
X    freeunit(np);
X}
X
Xprintlist(np)
XNODE *np;
X{
X    putchar('\n');
X    prln(np, 2);
X}
X
Xprln(np, indent)
XNODE *np;
X{
X    register NODE *svl, *nxtl;
X
X    for (svl=np; svl != NULL; svl = nxtl) {
X         nxtl = svl->n_next;
X         svl->n_next = NULL;
X         prnode(svl,indent);
X         svl->n_next = nxtl;
X         /* special hack for tag list */
X         if ((svl->n_flags & N_BRKPR) && svl->n_right)
X              prln(svl->n_right, indent+2);
X    }
X}
X
Xcodeprint(np)
XNODEP np;
X{
X    putchar('\n');
X    cprnode(np,0);
X}
X
Xcprnode(np,indent)
XNODE *np;
X{
X    int ni;
X    NODEP tp;
X
X    ni = indent+1;
X    while (indent--)
X         putchar(' ');
X    if (np == NULL) {
```

```
X         printf("<NULL>\n");
X         return;
X    }
X    put_nnm(np);  /* Note: BRKPR doesnt break long names */
X    if (np->g_offs)
X         printf(" o%ld ", np->g_offs);
X    if (np->g_rno)
X         printf(" r%d ", np->g_rno);
X    if (np->g_needs)
X         printf(" n%x ", np->g_needs);
X    if (debug) {
X         printf("@%lx ", np);
X         if (np->n_flags & N_COPYT)
X              printf("C ");
X         if (np->n_flags & N_BRKPR)
X              printf("B ");
X    }
X    if (np->n_flags & N_BRKPR) {
X         putchar('\n');
X         return;
X    }
X    if (np->g_betw)
X         printf(" {%s}", np->g_betw);
X    if (np->g_code) {
X         if (np->n_flags & N_COPYT)
X              printf(" <%s>", np->g_code);
X         else
X              for (tp=np->g_code; tp; tp = tp->g_code)
X                   printf(" <%s>", tp->n_name);
X    }
X    putchar(' ');
X    out_a(np, stdout);
X    putchar('\n');
X    if (np->n_left) {
X         cprnode(np->n_left,ni);
X    } else if (np->n_right)
X         cprnode(NULL, ni);
X    if (np->n_right) {
X         cprnode(np->n_right,ni);
X    }
X}
X
Xprintnode(np)
XNODE *np;
X{
X    putchar('\n');
X    prnode(np,0);
X}
X
Xprnode(np,indent)
XNODE *np;
X{
X    int ni;
```

```
X
X    ni = indent+1;
X    while (indent--)
X         putchar(' ');
X    if (np == NULL) {
X         printf("<NULL>\n");
X         return;
X    }
X    put_nnm(np);  /* Note: BRKPR doesnt break long names */
X    if (np->e_offs)
X         printf(" o%ld ", np->e_offs);
X    if (np->e_rno)
X         printf(" r%d ", np->e_rno);
X    if (np->e_fldw)
X         printf(" (%d,%d) ", np->e_fldw, np->e_fldo);
X    if (debug) {
X         printf("@%lx ", np);
X         if (np->n_flags & N_COPYT)
X              printf("C ");
X         if (np->n_flags & N_BRKPR)
X              printf("B ");
X    }
X    if (np->n_flags & N_BRKPR) {
X         putchar('\n');
X         return;
X    }
X    if (np->n_tptr) {
X         if (np->e_flags & 256) /* IMMEDID */
X              printf(" $$$ ");
X         tprint(np->n_tptr);
X    }
X    putchar('\n');
X    if (np->n_left) {
X         prnode(np->n_left,ni);
X    } else if (np->n_right)
X         prnode(NULL, ni);
X    if (np->n_right) {
X         prnode(np->n_right,ni);
X    }
X}
X
Xtprint(np)
XNODEP np;
X{
X    while (np != NULL) {
X         putchar(' ');
X         put_nnm(np);
X#ifdef HANS
X         if (np->t_size)
X              printf(" s%ld", np->t_size);
X         if (np->t_aln)
X              printf(" a%d", np->t_aln);
X#endif
```

```
X         if (debug)
X              printf("@%lx", np);
X         np = np->n_tptr;
X    }
X}
X
XNODEP
Xcopynode(op)
XNODEP op;
X{
X    NODEP np;
X
X    if (op == NULL) return NULL;
X    np = allocnode();
X    lcpy(np, op, NODELEN);
X    if (np->n_nmx)
X         np->n_nmx = copynode(np->n_nmx);
X    if (np->n_right)
X         np->n_right = copynode(np->n_right);
X    if (np->n_left)
X         np->n_left = copynode(np->n_left);
X    if (np->n_tptr)
X         np->n_flags |= N_COPYT;
X    return np;
X}
X
XNODEP
Xcopyone(op)
XNODEP op;
X{
X    NODEP np;
X
X    if (op == NULL) return NULL;
X    np = allocnode();
X    lcpy(np, op, NODELEN);
X    if (np->n_nmx)
X         np->n_nmx = copyone(np->n_nmx);
X    if (np->n_right)
X         np->n_right = NULL;
X    if (np->n_left)
X         np->n_left = NULL;
X    if (np->n_tptr)
X         np->n_flags |= N_COPYT;
X    return np;
X}
X
XNODEP
Xcopy_nol(op)
XNODEP op;
X{
X    NODEP np;
X
X    if (op == NULL) return NULL;
```

```
X    np = allocnode();
X    lcpy(np, op, NODELEN);
X    if (np->n_nmx)
X         np->n_nmx = copynode(np->n_nmx);
X    if (np->n_right) /* break right links */
X         np->n_right = NULL;
X    if (np->n_tptr)
X         np->n_flags |= N_COPYT;
X    return np;
X}
X
XNODEP
Xcopylist(np, tailp)
XNODE *np, **tailp;
X{
X    NODEP rv, nx;
X    register NODEP tail;
X
X    if (np == NULL) {
X         *tailp = NULL;
X         return NULL;
X    }
X    rv = copy_nol(np);
X    tail = rv;
X    while (tail->n_left) {
X         nx = copy_nol(tail->n_left);
X         tail->n_left = nx;
X         tail = nx;
X    }
X    *tailp = tail;
X    return rv;
X}
X
XNODE *
Xnthnode(np, n)
XNODE *np;
X{
X    while (n--)
X         if (np == NULL)
X              return NULL;
X         else
X              np=np->n_next;
X    return np;
X}
X
XNODE *
Xrthnode(np, n)
XNODE *np;
X{
X    while (n--)
X         if (np == NULL)
X              return NULL;
X         else
```

```
X             np=np->n_right;
X    return np;
X}
END_OF_FILE
if test 8680 -ne `wc -c <'hcc/NODES.C'`; then
    echo shar: \"'hcc/NODES.C'\" unpacked with wrong size!
fi
# end of 'hcc/NODES.C'
fi
if test -f 'top/DATA.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/DATA.C'\"
else
echo shar: Extracting \"'top/DATA.C'\" \(8563 characters\)
sed "s/^X//" >'top/DATA.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X
X/*
X * Routines for data flow analysis of a single instruction
X */
X
X#include "top.h"
X
X/*
X * idata
X *
X * For each instruction, we have some global information, as well
X * as flags indicating what the instruction does with its operands.
X * We need to know if each operand is set and/or referenced. If the
X * instruction has side-effects not directly related to its operands,
X * we need to know that as well, so "special case" code can deal with
X * that as well.
X */
Xstruct  idata    {
X
X    char iflag;        /* flags regarding the entire instruction */
X#define      SIDE 0x01 /* inst. has side-effects */
X#define      CC   0x02 /* inst. munges condition codes */
X
X    char op1f;         /* flags for the first and second operands */
X    char op2f;
X#define      SET  0x01 /* operand is set */
X#define      REF  0x02 /* operand is referenced */
X
X} idata[] =
```

```
X{
X    { CC,          REF,      REF|SET },    /* OR */
X    { CC,          REF,      REF },        /* BTST */
X    { 0,      REF,      SET },         /* MOVEP */
X    { CC,          REF,      REF|SET },    /* BCHG */
X    { CC,          REF,      REF|SET },    /* BCLR */
X    { CC,          REF,      REF|SET },    /* BSET */
X    { CC,          REF,      REF|SET },    /* AND */
X    { CC,          REF,      REF|SET },    /* SUB */
X    { CC,          REF,      REF|SET },    /* ADD */
X    { CC,          REF,      REF|SET },    /* EOR */
X    { CC,          REF,      REF },        /* CMP */
X    { CC,          REF,      SET },        /* MOVE */
X    { CC,          REF|SET,  0 },      /* NEGX */
X    { CC,          REF,      REF },        /* CHK */
X    { 0,      REF,      SET },        /* LEA */
X    { CC,          SET,      0 },      /* CLR */
X    { CC,          REF|SET,  0 },      /* NEG */
X    { CC,          REF|SET,  0 },      /* NOT */
X    { CC,          REF|SET,  0 },      /* NBCD */
X    { CC,          REF|SET,  0 },      /* SWAP */
X    { SIDE,        REF,      0 },      /* PEA */
X    { CC,          REF|SET,  0 },      /* EXT */
X    { SIDE,        REF,      SET },        /* MOVEM */
X    { CC,          REF,      0 },      /* TST */
X    { CC,          REF|SET,  0 },      /* TAS */
X    { 0,      REF,      0 },      /* TRAP */
X    { SIDE,        REF|SET,  REF },        /* LINK */
X    { SIDE,        REF|SET,  0 },      /* UNLK */
X    { 0,      0,        0 },      /* RESET */
X    { 0,      0,        0 },      /* NOP */
X    { CC,          REF,      0 },      /* STOP */
X    { SIDE|CC,     0,        0 },      /* RTE */
X    { SIDE,        0,        0 },      /* RTS */
X    { 0,      0,        0 },      /* TRAPV */
X    { SIDE|CC,     0,        0 },      /* RTR */
X    { SIDE,        REF,      0 },      /* JSR */
X    { 0,      REF,      0 },      /* JMP */
X    { CC,          REF,      REF|SET },    /* ADDQ */
X    { 0,      SET,      0 },      /* ST */
X    { CC,          REF|SET,  REF },        /* DBT */
X    { CC,          REF,      REF|SET },    /* SUBQ */
X    { 0,      SET,      0 },      /* SF */
X    { CC,          REF|SET,  REF },        /* DBRA (dbf) */
X    { 0,      SET,      0 },      /* SHI */
X    { CC,          REF|SET,  REF },        /* DBHI */
X    { 0,      SET,      0 },      /* SLS */
X    { CC,          REF|SET,  REF },        /* DBLS */
X    { 0,      SET,      0 },      /* SCC */
X    { CC,          REF|SET,  REF },        /* DBCC */
X    { 0,      SET,      0 },      /* SCS */
X    { CC,          REF|SET,  REF },        /* DBCS */
X    { 0,      SET,      0 },      /* SNE */
```

```
X    { CC,          REF|SET,  REF },        /* DBNE */
X    { 0,      SET,      0 },      /* SEQ */
X    { CC,          REF|SET,  REF },        /* DBEQ */
X    { 0,      SET,      0 },      /* SVC */
X    { CC,          REF|SET,  REF },        /* DBVC */
X    { 0,      SET,      0 },      /* SVS */
X    { CC,          REF|SET,  REF },        /* DBVS */
X    { 0,      SET,      0 },      /* SPL */
X    { 0,      SET,      0 },      /* SMI */
X    { CC,          REF|SET,  REF },        /* DBMI */
X    { 0,      SET,      0 },      /* SGE */
X    { CC,          REF|SET,  REF },        /* DBGE */
X    { 0,      SET,      0 },      /* SLT */
X    { CC,          REF|SET,  REF },        /* DBLT */
X    { 0,      SET,      0 },      /* SGT */
X    { CC,          REF|SET,  REF },        /* DBGT */
X    { 0,      SET,      0 },      /* SLE */
X    { CC,          REF|SET,  REF },        /* DBLE */
X    { 0,      REF,      0 },      /* BRA */
X    { SIDE,        REF,      0 },      /* BSR */
X    { 0,      REF,      0 },      /* BHI */
X    { 0,      REF,      0 },      /* BLS */
X    { 0,      REF,      0 },      /* BCC */
X    { 0,      REF,      0 },      /* BCS */
X    { 0,      REF,      0 },      /* BNE */
X    { 0,      REF,      0 },      /* BEQ */
X    { 0,      REF,      0 },      /* BVC */
X    { 0,      REF,      0 },      /* BVS */
X    { 0,      REF,      0 },      /* BPL */
X    { 0,      REF,      0 },      /* BMI */
X    { 0,      REF,      0 },      /* BGE */
X    { 0,      REF,      0 },      /* BLT */
X    { 0,      REF,      0 },      /* BGT */
X    { 0,      REF,      0 },      /* BLE */
X    { CC,          REF,      SET },        /* MOVEQ */
X    { CC,          REF,      REF|SET },    /* DIVU */
X    { CC,          REF,      REF|SET },    /* SBCD */
X    { CC,          REF,      REF|SET },    /* DIVS */
X    { CC,          REF,      REF|SET },    /* SUBX */
X    { CC,          REF,      REF },        /* CMPM */
X    { CC,          REF,      REF|SET },    /* MULU */
X    { CC,          REF,      REF|SET },    /* ABCD */
X    { 0,      REF|SET,  REF|SET },    /* EXG */
X    { CC,          REF,      REF|SET },    /* MULS */
X    { CC,          REF,      REF|SET },    /* ADDX */
X    { CC,          REF,      REF|SET },    /* ASR */
X    { CC,          REF,      REF|SET },    /* LSR */
X    { CC,          REF,      REF|SET },    /* ROXR */
X    { CC,          REF,      REF|SET },    /* ROR */
X    { CC,          REF,      REF|SET },    /* ASL */
X    { CC,          REF,      REF|SET },    /* LSL */
X    { CC,          REF,      REF|SET },    /* ROXL */
X    { CC,          REF,      REF|SET },    /* ROL */
```

```
X    { 0,      0,        0 },      /* DC */
X};
X
X/*
X * chkset(op) - check to see if operand 'op' sets a register
X *
X * This given operand is set by an instruction. Depending on the
X * addressing mode used, this may set a register. If so, return
X * an appropriate mask. This only happens with register direct
X * addressing.
X */
Xint
Xchkset(op)
Xregister struct   opnd *op;
X{
X    switch (M(op->amode)) {
X
X    case REG:
X         return RM(op->areg);
X    case REGI:
X         if (op->amode & (INC|DEC))
X              return RM(op->areg);
X         else
X              return 0;
X
X    default:
X         return 0;
X    }
X}
X
X/*
X * chkref(op) - check to see if operand 'op' references a register
X *
X * Checks for register references in source or destination
X * operands, since they can occur in either.
X */
Xint
Xchkref(op, is_src)
Xregister struct   opnd *op;
Xregister bool     is_src;       /* is the operand a source? */
X{
X    switch (M(op->amode)) {
X
X    case NONE:
X    case IMM:
X    case ABS:
X    case PCD:
X         return 0;
X
X    case REG:
X         if (is_src)
X              return RM(op->areg);
X         else
```

```
X             return 0;
X
X    case REGI:
X    case REGID:
X         return RM(op->areg);
X
X    case REGIDX:
X         return (RM(op->areg) | RM(op->ireg));
X
X    case PCDX:
X         return RM(op->ireg);
X
X    default:
X         fprintf(stderr, "illegal mode in chkref() %d\n", M(op->amode));
X         exit(1);
X    }
X}
X
X/*
X * chkside(ip, type) - check for side-effects of 'ip'
X *
X * Return a mask of registers set or referenced (depending on 'type')
X * by the given instruction. For example, "pea" sets and references
X * the stack pointer.
X */
Xint
Xchkside(ip, type)
XINST    *ip;
Xint type;
X{
X    switch (ip->opcode) {
X    case PEA:     /* refs/sets the stack pointer */
X         return RM(SP);
X
X    case LINK:         /* refs/sets SP */
X         return RM(SP);
X
X    case UNLK:
X         if (type == SET)
X              return RM(SP);
X         else
X              return 0;
X
X    case RTE:
X    case RTS:
X    case RTR:
X         if (type == SET)
X              return RM(SP);
X         else
X              return RM(SP)|RM(D0);   /* return value's in D0 */
X
X    case JSR:
X    case BSR:
```

```
X         /*
X          * We have to account, here, for what the called
X          * routine might do. Registers D0-3 and A0-3 may
X          * be munged.
X          */
X         if (type == SET)
X              return    RM(A0)|RM(A1)|RM(A2)|
X                   RM(D0)|RM(D1)|RM(D2)|RM(SP);
X         else
X              return    RM(SP);
X
X
X    case MOVEM:
X         /*
X          * We should really check for a register mask spec.
X          * here and parse it. The simple solution is to assume
X          * that all the registers used for register variables
X          * are referenced or modified.
X          */
X         return RM(A3)|RM(A4)|RM(A5)|RM(A7)|
X                RM(D3)|RM(D4)|RM(D5)|RM(D6)|RM(D7);
X
X    default:
X         fprintf(stderr, "chkside() - unknown opcode\n");
X         exit(1);
X    }
X}
X
X/*
X * reg_set(ip) - return mask of regs set by 'ip'
X */
Xint
Xreg_set(ip)
Xregister INST     *ip;
X{
X    int  mask = 0; /* build up a register mask */
X
X    if (idata[ip->opcode].op1f & SET)
X         mask |= chkset(&ip->src);
X    if (idata[ip->opcode].op1f & REF) {
X         if ((ip->src.amode & (INC|DEC)) != 0)
X              mask |= RM(ip->src.areg);
X    }
X
X    if (idata[ip->opcode].op2f & SET)
X         mask |= chkset(&ip->dst);
X    if (idata[ip->opcode].op2f & REF) {
X         if ((ip->dst.amode & (INC|DEC)) != 0)
X              mask |= RM(ip->dst.areg);
X    }
X
X    if (idata[ip->opcode].iflag & SIDE)
X         mask |= chkside(ip, SET);
```

```
X
X    return mask;
X}
X
X/*
X * reg_ref(ip) - return mask of regs referenced by 'ip'
X */
Xint
Xreg_ref(ip)
Xregister INST     *ip;
X{
X    int  mask = 0; /* build up a register mask */
X
X    mask |= chkref(&ip->src, idata[ip->opcode].op1f & REF);
X    mask |= chkref(&ip->dst, idata[ip->opcode].op2f & REF);
X    if (idata[ip->opcode].iflag & SIDE)
X         mask |= chkside(ip, REF);
X
X    return mask;
X}
X
X/*
X * sets(ip, reg) - is 'reg' set by the instruction 'ip'?
X */
Xbool
Xsets(ip, reg)
XINST    *ip;
Xint reg;
X{
X    return ((reg_set(ip) & RM(reg)) != 0);
X}
X
X/*
X * refs(ip, reg) - is 'reg' referenced by the instruction 'ip'?
X */
Xbool
Xrefs(ip, reg)
XINST    *ip;
Xint reg;
X{
X    return ((reg_ref(ip) & RM(reg)) != 0);
X}
X
X/*
X * uses(ip, ref) - is 'reg' used by the instruction 'ip'?
X */
Xbool
Xuses(ip, reg)
XINST    *ip;
Xint reg;
X{
X    return sets(ip, reg) || refs(ip, reg);
X}
```

```
END_OF_FILE
if test 8563 -ne `wc -c <'top/DATA.C'`; then
    echo shar: \"'top/DATA.C'\" unpacked with wrong size!
fi
# end of 'top/DATA.C'
fi
echo shar: End of archive 3 \(of 9\).
cp /dev/null ark3isdone
MISSING=""
for I in 1 2 3 4 5 6 7 8 9 ; do
    if test ! -f ark${I}isdone ; then
     MISSING="${MISSING} ${I}"
    fi
done
if test "${MISSING}" = "" ; then
    echo You have unpacked all 9 archives.
    rm -f ark[1-9]isdone ark[1-9][0-9]isdone
else
    echo You still need to unpack the following archives:
    echo "        " ${MISSING}
fi
##  End of shell archive.
exit 0
```

```
From 6317641913485363179
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,d6fd9f40eda8db51,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-24 18:47:55 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!ames!apple!sun-barr!newstop!sun!
grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i095: sozobon1.2 -- Update to Sozobon C compiler  part04/09
Keywords: shar, C
Message-ID: <34606@grapevine.uucp>
Date: 25 Oct 89 01:47:55 GMT
Sender: news@grapevine.EBay
Reply-To: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Lines: 2603
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Posting-number: Volume 2, Issue 95
Archive-name: sozobon1.2/part04

#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
```

```
#        "End of archive 4 (of 9)."
# Contents:  hcc/D2.C hcc/DECL.C hcc/EXPR.C hcc/TOK.C
# Wrapped by koreth@panarthea on Tue Oct 24 18:40:45 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'hcc/D2.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/D2.C'\"
else
echo shar: Extracting \"'hcc/D2.C'\" \(9938 characters\)
sed "s/^X//" >'hcc/D2.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  d2.c
X *
X *  Declaration subroutines
X *
X *  Mostly routines for initializations
X */
X
X#include <stdio.h>
X#include "param.h"
X#include "tok.h"
X#include "nodes.h"
X#include "cookie.h"
X
X#if MMCC
Xoverlay "pass2"
X#endif
X
Xextern NODEP cur;
Xextern NODEP symtab[];
Xextern level;
X
Xextern int oflags[];
X#define debugi    oflags['i'-'a']
X
Xsu_size(lp, cp, xp, isunion)
Xregister long *lp;
Xchar *cp;
Xregister NODE *xp;
X{
X    long sz;
X    char al;
X
X    sz = xp->n_tptr->t_size;
X    al = xp->n_tptr->t_aln;
```

```
X    if (isunion) {
X         *lp = *lp > sz ? *lp : sz;
X    } else {
X         while (al & (*lp)) {    /* align new field */
X              (*lp)++;
X              xp->e_offs++;
X         }
X         *lp += sz;
X    }
X    *cp = *cp > al ? *cp : al;
X}
X
Xlc_size(lp, rp, xp)
Xregister long *lp;
Xint *rp;
Xregister NODE *xp;
X{
X    long sz;
X    char al;
X    long arg_size();
X#ifdef LAT_HOST
X    long tsz;
X#endif
X
X    if (level > 1 && xp->e_sc == K_REGISTER) {
X         if (lc_reg(rp, xp))
X              return;
X         else
X              xp->e_sc = K_AUTO;
X    }
X    if (xp->e_sc == K_AUTO || level == 1) {
X         sz = xp->n_tptr->t_size;
X         al = xp->n_tptr->t_aln;
X         while (al & (*lp)) {    /* align new field */
X              (*lp)++;
X              xp->e_offs++;
X         }
X         if (level == 1) {
X#ifndef LAT_HOST
X              sz = arg_size(sz,xp);
X#else
X              tsz = arg_size(sz,xp);
X              sz = tsz;
X#endif
X              xp->e_offs += ARG_BASE + *lp;
X         }
X         *lp += sz;
X         if (level != 1)
X              xp->e_offs = LOC_BASE - *lp;
X    }
X}
X
Xsu_fld(lp, alp, xp, fldw, fop)
```

```
Xregister long *lp;
Xchar *alp;
Xregister NODE *xp;
Xint *fop;
X{
X    if (*alp < ALN_I)
X         *alp = ALN_I;
X    if (fldw == 0) {
X         afterfld(lp, fop);
X         return;
X    }
X    if (fldw + *fop > 8*SIZE_I)
X         afterfld(lp, fop);
X    if (xp) {
X         xp->e_fldw = fldw;
X         xp->e_fldo = *fop;
X    }
X    *fop += fldw;
X}
X
Xafterfld(szp, fop)
Xlong *szp;
Xint *fop;
X{
X    if (*fop) {
X         *szp += SIZE_I;
X         *fop = 0;
X    }
X}
X
Xok_gsh(sc, np)
XNODE *np;
X{
X    if (sc == K_REGISTER || sc == K_AUTO) {
X         error("reg/auto outside fun");
X         return 0;
X    }
X    return ok_ty(np, NULL);
X}
X
Xok_gx(np, endp)
XNODEP np, endp;
X{
X    if (np)
X         return ok_ty(np->n_tptr, endp);
X    return 0;
X}
X
Xok_lsh(sc, np)
XNODE *np;
X{
X    return ok_ty(np, NULL);
X}
```

```
X
Xarytoptr(np)
XNODEP np;
X{
X    NODEP tp = np->n_tptr;
X    NODEP copyone();
X
X    if (np->n_flags & N_COPYT) {     /* cant change if a dupl. */
X         tp = copyone(tp);
X         np->n_tptr = tp;
X         np->n_flags &= ~N_COPYT;
X    }
X    tp->t_token = STAR;
X    strcpy(tp->n_name, "Aptr to");
X}
X
Xok_lx(np,endp)
XNODEP np, endp;
X{
X    if (np) {
X         if (level == 1 && np->n_tptr->t_token == '[')
X              arytoptr(np);
X         return ok_ty(np->n_tptr, endp);
X    }
X    return 0;
X}
X
Xok_suh(np)
XNODEP np;
X{
X    return 1;
X}
X
Xok_sux(np, endp)
XNODEP np, endp;
X{
X    if (np)
X         return ok_ty(np->n_tptr, endp);
X    return 0;
X}
X
Xok_enx(np, endp)
XNODEP np, endp;
X{
X    if (np && np->n_tptr == endp)    /* no modifiers */
X         return 1;
X    return 0;
X}
X
Xok_cast(np, endp)
XNODEP np, endp;
X{
X    if (np)
```

```
X         return ok_ty(np, endp);
X    return 0;
X}
X
Xok_ty(np, endp)
Xregister NODEP np, endp;
X{
X    NODEP child;
X    long csize;
X    long conlval();
X
X    if (np == endp)
X         return 1;
X    child = np->n_tptr;
X    if (child) {
X         if (ok_ty(child, endp) == 0)
X              return 0;
X         csize = child->t_size;
X    }
X
X    switch (np->t_token) {
X    case STAR:
X         np->t_size = SIZE_P;
X         np->t_aln = ALN_P;
X         break;
X    case '(':
X         /* size 0 okay - fun ret void */
X         if (child->t_token == '[') {
X              error("bad func");
X              return 0;
X         }
X         /* size 0 */
X         break;
X    case '[':
X         if (csize == 0) {
X              error("bad array");
X              return 0;
X         }
X         if (np->n_right) {
X              csize *= conlval(np->n_right);
X              np->n_right = NULL;
X              np->t_size = csize;
X         }
X         np->t_aln = child->t_aln;
X         break;
X    default:
X         return 1;
X    }
X    return 1;
X}
X
Xok_revx(rv,forcast)
XNODEP rv;
```

```
X{
X    if (rv == NULL)
X         return 1;
X    if (forcast == 0 && rv->e_token != ID) {
X         error("need ID");
X         return 0;
X    }
X    if (forcast && rv->e_token == ID) {
X         error("ID in cast");
X         return 0;
X    }
X    return 1;
X}
X
Xopt_ginit(xp)
XNODEP xp;
X{
X    if (xp->e_token != ID)
X         return;
X    if (xp->n_tptr->t_token == '(')
X         return;
X    switch (xp->e_sc) {
X    case K_STATIC:
X    case HERE_SC:
X         if (cur->e_token == '=') {
X              out_gv(xp, 0);
X              fadvnode();
X              g_init(xp->n_tptr);
X         } else
X              out_gv(xp, 1);
X    }
X}
X
Xopt_linit(xp)
XNODEP xp;
X{
X    if (xp->e_token != ID)
X         return;
X    if (xp->n_tptr->t_token == '(')
X         return;
X    switch (xp->e_sc) {
X    case K_STATIC:
X         if (cur->e_token == '=') {
X              out_gv(xp, 0);
X              fadvnode();
X              g_init(xp->n_tptr);
X         } else
X              out_gv(xp, 1);
X         to_text();
X         break;
X    case K_AUTO:
X    case K_REGISTER:
X         if (cur->e_token == '=')
```

```
X             a_init(xp);
X         break;
X    }
X}
X
Xa_init(op)
XNODEP op;
X{
X    register NODEP np, xp;
X    NODEP assignx(), copynode();
X
X    np = cur;  advnode();
X    xp = assignx();
X    op = copynode(op);
X    np->n_left = op;
X    np->n_right = xp;
X    np->e_type = E_BIN;
X    do_expr(np, FORSIDE);
X    return;
X}
X
Xopt_enval(intp)
Xint *intp;
X{
X    NODEP np;
X    NODEP questx();
X
X    if (cur->e_token == '=') {
X         fadvnode();
X         np = questx();
X         *intp = conxval(np);
X         return;
X    }
X}
X
Xopt_field(xp,wdp,isunion)
XNODE *xp;
Xint *wdp;
X{
X    NODEP np;
X    NODEP questx();
X    int i;
X
X    *wdp = -1;
X    if (isunion) return;
X    if (cur->e_token == ':') {
X         fadvnode();
X         np = questx();
X         i = conxval(np);
X         if (i > 8*SIZE_I) {
X              error("field too big");
X              i = 8*SIZE_I;
X         }
```

```
X        if (xp) {
X             if (i <= 0 || bad_fty(xp->n_tptr)) {
X                  error("bad field");
X                  return;
X             }
X        } else if (i < 0) {
X             error("neg field width");
X             return;
X        }
X        *wdp = i;
X        return;
X    }
X}
X
Xbad_fty(tp)
XNODEP tp;
X{
X    int tok;
X
X    tok = tp->t_token;
X    if (tok == K_INT || tok == K_UNSIGNED)
X         return 0;
X    return 1;
X}
X
Xfield(xp, wd, ofp)
XNODEP xp;
Xint *ofp;
X{
X}
X
XNODEP
Xdef_type()
X{
X    NODEP bas_type();
X
X    return bas_type(K_INT);
X}
X
X#define NSC  LAST_SC-FIRST_SC+1
X#define NBAS LAST_BAS-FIRST_BAS+1
X
XNODE basics[NBAS];
XNODE str_ptr, fun_int;
X
Xstruct bt {
X    char *name;
X    int  size;
X    char align;
X} btbl[] = {
X    {"Uchar",SIZE_C, ALN_C},
X    {"Ulong",SIZE_L, ALN_L},
X    {"Long", SIZE_L,  ALN_L},
```

```
X    {"Short",SIZE_S, ALN_S},
X    {"Uns",       SIZE_U, ALN_U},
X    {"Int",       SIZE_I, ALN_I},
X    {"Char", SIZE_C, ALN_C},
X    {"Float",SIZE_F, ALN_F},
X    {"Dbl",       SIZE_D, ALN_D},
X    {"Void", 0},
X};
X
XNODEP
Xbas_type(btype)
X{
X    NODEP rv;
X    static once = 0;
X
X    if (once == 0) {
X         once++;
X
X         sprintf(str_ptr.n_name, "Ptr to");
X         str_ptr.t_token = STAR;
X         str_ptr.n_tptr = bas_type(K_CHAR);
X         str_ptr.n_flags = N_COPYT;
X         str_ptr.t_size = SIZE_P;
X         str_ptr.t_aln = ALN_P;
X
X         sprintf(fun_int.n_name, "Fun ret");
X         fun_int.t_token = '(';
X         fun_int.n_tptr = bas_type(K_INT);
X         fun_int.n_flags = N_COPYT;
X    }
X    if (btype == SCON)
X         return &str_ptr;
X    else if (btype == '(')
X         return &fun_int;
X    rv = &basics[btype-FIRST_BAS];
X    if (rv->t_token == 0) {
X         rv->t_token = btype;
X         rv->t_size = btbl[btype-FIRST_BAS].size;
X         rv->t_aln = btbl[btype-FIRST_BAS].align;
X         sprintf(rv->n_name, btbl[btype-FIRST_BAS].name);
X    }
X    return rv;
X}
X
X/* new function name seen in expr */
XNODEP
Xnew_fun(op)
XNODE *op;
X{
X    NODEP np;
X    NODEP copyone();
X
X    /* we know left, right and tptr are NULL */
```

```
X    np = copyone(op); /* ID node */
X    np->n_tptr = bas_type('(');
X    np->n_flags = N_COPYT;
X    np->e_sc = K_EXTERN;
X    new_sym(symtab, np);
X    return np;
X}
X
X/* declare arg name as int */
Xdef_arg(listpp, op)
XNODE **listpp, *op;
X{
X    register NODEP np;
X    NODEP copyone();
X
X    np = copyone(op);
X    np->n_tptr = bas_type(K_INT);
X    np->n_flags = N_COPYT;
X    np->e_sc = K_AUTO;
X    new_sym(listpp, np);
X}
X
X/* initialize 0 or 1 thing of any type (tp) */
Xg_init(tp)
Xregister NODEP tp;
X{
X    int nsee;
X    long sz;
X    int oldsize;
X    int seebr = 0;
X
X    if (cur->e_token == SCON &&
X            tp->t_token == '[' &&
X            tp->n_tptr->t_token == K_CHAR) { /* hack for SCON ary */
X              nsee = out_scon(cur);
X              fadvnode();
X              a_fix(tp, nsee);
X              return 1;
X    }
X
X    if (cur->e_token == '{') {
X         fadvnode();
X         seebr = 1;
X    }
X
X    switch (tp->t_token) {
X    case '[':
X         if (tp->t_size)
X              oldsize = tp->t_size / tp->n_tptr->t_size;
X         else
X              oldsize = 0;
X         nsee = inita(tp->n_tptr, oldsize);
X         if (nsee)
```

```
X             a_fix(tp, nsee);
X         break;
X    case K_STRUCT:
X         o_aln(tp->t_aln);
X         nsee = inits(tp->n_right);
X         break;
X    case K_UNION:
X         o_aln(tp->t_aln);
X         nsee = g_init(tp->n_right->n_tptr);
X         if (nsee) {
X              sz = tp->t_size - tp->n_right->n_tptr->t_size;
X              if (sz)
X                   o_nz(sz, 0);
X         }
X         break;
X    default:
X         nsee = init1(tp);
X         break;
X    }
X
X    if (seebr) {
X         if (cur->e_token == ',')
X              fadvnode();
X         eat('}');
X    }
X    return nsee ? 1 : 0;
X}
X
X/* initialize one (or 0) scalar to an expr */
Xinit1(tp)
Xregister NODEP tp;
X{
X    NODEP xp;
X    NODEP assignx();
X
X    if (debugi) {
X         printf("init1");
X         printnode(tp);
X    }
X    xp = assignx();
X    if (xp) {
X         if (debugi)
X              printnode(xp);
X         o_vinit(tp, xp);
X         return 1;
X    } else
X         return 0;
X}
X
X/* set array size or fill array with zeros */
Xa_fix(tp, nsee)
Xregister NODEP tp;
X{
```

```
X    int oldsize;
X
X    if (tp->t_size) {
X         oldsize = tp->t_size / tp->n_tptr->t_size;
X         if (oldsize > nsee) {
X              o_nz(tp->n_tptr->t_size * (oldsize-nsee),
X                   tp->n_tptr->t_aln);
X         } else if (oldsize < nsee) {
X              error("too many init exprs");
X         }
X    } else
X         tp->t_size = nsee * tp->n_tptr->t_size;
X}
X
X/* initialize up to max items of type tp */
X/* if max is 0, any number is okay */
X
Xinita(tp, maxi)
XNODEP tp;
X{
X    int nsee;
X
X    nsee = g_init(tp);
X    if (nsee == 0)
X         return 0;
X
X    while (cur->e_token == ',') {
X         if (nsee == maxi)
X              break;
X         fadvnode();
X         nsee += g_init(tp);
X    }
X    return nsee;
X}
X
X/* initialize (possible) structure */
Xinits(np)
Xregister NODEP np;
X{
X    int see1;
X
X    see1 = g_init(np->n_tptr);
X    if (see1 == 0)
X         return 0;
X
X    while (np->n_next) {
X         np = np->n_next;
X         if (cur->e_token == ',') {
X              fadvnode();
X              see1 = g_init(np->n_tptr);
X         } else
X              see1 = 0;
X         if (see1 == 0)
```

```
X             z_init(np->n_tptr);
X    }
X
X    return 1;
X}
X
Xz_init(tp)
Xregister NODEP tp;
X{
X    switch (tp->t_token) {
X    case '[':
X    case K_STRUCT:
X    case K_UNION:
X         o_nz(tp->t_size, tp->t_aln);
X         break;
X    default:
X         out_zi(tp);
X    }
X}
END_OF_FILE
if test 9938 -ne `wc -c <'hcc/D2.C'`; then
    echo shar: \"'hcc/D2.C'\" unpacked with wrong size!
fi
# end of 'hcc/D2.C'
fi
if test -f 'hcc/DECL.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/DECL.C'\"
else
echo shar: Extracting \"'hcc/DECL.C'\" \(10944 characters\)
sed "s/^X//" >'hcc/DECL.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  decl.c
X *
X *  Do all declarations
X *
X *  Currently,
X *       struct tags are local
X *       struct members are tied to the struct
X *       enum tags are ignored
X *       enum members are local
X */
X
X#include <stdio.h>
X#include "param.h"
```

```
X#include "tok.h"
X#include "nodes.h"
X
Xextern NODE *cur;
Xextern level;
X
XNODEP symtab[NHASH], tagtab;
Xextern NODE *blktab;
X
XNODEP alltags(), allsyms(), llook(), hlook();
X
Xextern int oflags[];
X#define debug    oflags['v'-'a']
X
X/* look for global data decls
X   return when see something weird
X   return last ID declared */
XNODEP
Xglb_decls()
X{
X   register NODEP head, xp;
X   NODEP d_type(), def_type(), d_declr();
X   int sclass;
X
X   for(;;) {
X        sclass = d_scl(HERE_SC);
X        head = d_type();
X        if (head == NULL)
X             head = def_type();
X        if (ok_gsh(sclass, head) == 0)
X             continue;
X   more:
X        xp = d_declr(head,0);
X        if (ok_gx(xp,head)) {
X             xp->e_sc = sclass;
X             opt_ginit(xp);
X             new_sym(symtab,xp);
X             if (xp->n_tptr->t_token == '(') {    /* func */
X                  if (cur->e_token == ',' ||
X                      cur->e_token == ';')
X                       fix_fun(xp);
X                  else
X                       return xp;
X             }
X        }
X
X        if (cur->e_token == ',') {
X             fadvnode();
X             goto more;
X        }
X
X        if (cur->e_token == ';') {
X             fadvnode();
```

```
X        } else
X             return NULL;
X    }
X}
X
X/* do local or arg decls
X   return 1 if see something */
Xloc_decls()
X{
X    register NODEP head, xp;
X    NODEP d_type(), def_type(), d_declr();
X    int sclass;
X    int regs;
X    long size;
X    int rv = 0;
X
X    size = level > 2 ? blktab->n_next->b_size : 0;
X    regs = level > 1 ? blktab->n_next->b_regs : 0;
X    while (is_ty_start()) {
X         rv++;
X         sclass = d_scl(K_AUTO);
X         head = d_type();
X         if (head == NULL)
X              head = def_type();
X         if (ok_lsh(sclass, head) == 0)
X              continue;
X    more:
X         xp = d_declr(head,0);
X         if (ok_lx(xp,head)) {
X              xp->e_sc = sclass;
X              if (level > 1) {   /* not args */
X                   lc_size(&size, &regs, xp);
X                   out_advice(xp);
X              }
X              new_sym(&blktab->b_syms,xp);
X              fix_fun(xp);
X              opt_linit(xp,sclass);
X         }
X
X         if (cur->e_token == ',') {
X              fadvnode();
X              goto more;
X         }
X
X         if (cur->e_token == ';') {
X              fadvnode();
X         } else {
X              error("expect ;");
X              return 1;
X         }
X    }
X    while (STACKALN & size)
X         size++;
```

```
X    blktab->b_size = size;
X    blktab->b_regs = regs;
X    return rv;
X}
X
X/* Decls inside Struct/Union */
Xsu_decls(listpp, isunion, sizep, alnp)
XNODEP *listpp;
Xlong *sizep;
Xchar *alnp;
X{
X    register NODEP head, xp;
X    NODEP d_type(), d_declr();
X    long size;
X    char aln;
X    int fldw, fldoff;
X
X    aln = 0;
X    size = 0;
X    fldoff = 0;
X    for(;;) {
X         head = d_type();
X         if (head == NULL)
X              goto out;
X         if (ok_suh(head) == 0)
X              continue;
X    more:
X         xp = d_declr(head,0);
X         opt_field(xp,&fldw,isunion);
X         if (ok_sux(xp,head)) {
X              if (fldw > 0) {    /* handle field */
X                   su_fld(&size,&aln,xp,fldw,&fldoff);
X                   xp->e_offs = size;
X              } else {      /* handle non-field */
X                   afterfld(&size,&fldoff);
X                   xp->e_offs = isunion ? 0 : size;
X                   su_size(&size,&aln,xp,isunion);
X              }
X              new_sym(listpp,xp);
X              listpp = &xp->n_next;
X         } else if (fldw == 0) {
X              afterfld(&size, &fldoff);
X         }
X
X         if (cur->e_token == ',') {
X              fadvnode();
X              goto more;
X         }
X
X         if (cur->e_token == ';') {
X              fadvnode();
X         } else
X              goto out;
```

```
X    }
Xout:
X    afterfld(&size,&fldoff);
X    while (aln & size)
X         size++;
X    *sizep = size;
X    *alnp = aln;
X    return;
X}
X
X/* Decls inside Enum */
Xen_decls()
X{
X    register NODEP head, xp;
X    NODEP bas_type(), d_declr();
X    int curval = 0;
X
X    for(;;) {
X         head = bas_type(K_INT);
X    more:
X         xp = d_declr(head,0);
X         if (ok_enx(xp,head)) {
X              opt_enval(&curval);
X              xp->e_ival = curval++;
X              xp->e_sc = ENUM_SC;
X              new_sym(level ? blktab->b_syms : (NODE *)symtab,
X                   xp);
X         }
X
X         if (cur->e_token == ',') {
X              fadvnode();
X              goto more;
X         }
X
X         return;
X    }
X}
X
X/*
X * called from expr.c, make a cast
X * only called if is_ty_start();
X */
XNODE *
Xmakecast()
X{
X    NODEP head, xp;
X    register NODEP np;
X    NODEP d_type(), d_declr(), def_type();
X
X    head = d_type();   /* we know this is not NULL */
X    xp = d_declr(head, 1);
X    if (ok_cast(xp,head) == 0) {
X         xp = def_type();   /* return cast to INT */
```

```
X    }
X    np = allocnode();
X    np->e_token = TCONV;
X    np->n_tptr = xp;
X    if (xp == head)
X         np->n_flags |= N_COPYT;
X    if (debug) {
X         printf("Make cast");
X         printnode(np);
X    }
X    return np;
X}
X
Xis_ty_start()
X{
X    NODEP rv;
X
X    if (is_tykw(cur->e_token))
X         return 1;
X    if (cur->e_token == ID) {
X         rv = allsyms(cur);
X         if (rv && rv->e_sc == K_TYPEDEF)
X              return 1;
X    }
X    return 0;
X}
X
X/* assemble decl and put in listpp */
Xnew_sym(listpp, xp)
XNODEP *listpp;
XNODEP xp;
X{
X    NODEP old;
X
X    if (xp == NULL)
X         return 0;
X/* put in table */
X    if (debug) {
X         printf("New sym sc %c", "EARTSCH"[xp->e_sc-K_EXTERN]);
X         printnode(xp);
X    }
X    /* later look for previous definition */
X    if (listpp == (NODE **)symtab) {
X         old = hlook(listpp, xp);
X         if (old == NULL || def2nd(old, xp))
X              puthlist(listpp, xp);
X    } else {
X         old = llook(*listpp, xp);
X         if (old == NULL || def2nd(old, xp))
X              putlist(listpp, xp);
X    }
X    return 1;
X}
```

```
X
X/* look for storage class */
Xd_scl(defau)
X{
X    int rv;
X
X    if (is_sclass(cur->e_token)) {
X         rv = cur->e_token;
X         fadvnode();
X         return rv;
X    }
X    /* no storage class specified */
X    return defau;
X}
X
XNODEP
Xd_declr(head, forcast)
XNODEP head;
X{
X    NODEP e1;
X    NODEP declarator(), rev_decl();
X    NODEP xp, tailp;
X
X    e1 = declarator();
X    xp = rev_decl(e1, &tailp, forcast);
X    if (xp) {
X         tailp->n_tptr = head;
X         tailp->n_flags |= N_COPYT;
X         return xp;
X    } else if (forcast)
X         return head;
X    else
X         return NULL;
X}
X
XNODEP
Xrev_decl(np,tailpp,forcast)
XNODEP np, *tailpp;
X{
X    NODEP rv, scan, nxt;
X
X    rv = NULL;
X    for (scan = np; scan != NULL; scan = nxt) {
X         nxt = scan->n_next;
X         scan->n_next = NULL;
X         if (rv == NULL) {
X              *tailpp = scan;
X              scan->n_tptr = NULL;
X              rv = scan;
X         } else {
X              scan->n_tptr = rv;
X              rv = scan;
X         }
```

```
X         e_to_t(rv);
X         switch (rv->t_token) {
X         case UNARY '*':
X              sprintf(rv->n_name, "Ptr to");
X              break;
X         case '(':
X              sprintf(rv->n_name, "Fun ret");
X              break;
X         case '[':
X              sprintf(rv->n_name, "Ary of");
X              break;
X         case ID:
X              break;
X         default:
X              error("bad type xpr");
X              return NULL;
X         }
X    }
X    /* if normal decl and see something, must see id first */
X    if (!ok_revx(rv,forcast))
X         rv = NULL;
X    return rv;
X}
X
X/*
X * Looking for type part of a decl
X */
XNODEP
Xd_type()
X{
X    int btype, adj;
X    NODEP rv;
X    NODEP bas_type(), decl_su(), decl_enum();
X
X    /* look for 'struct', 'union', 'enum' or typedef ID */
X    switch (cur->e_token) {
X    case ID:
X         rv = allsyms(cur);
X         if (rv && rv->e_sc == K_TYPEDEF) {
X              fadvnode();
X              rv = rv->n_tptr;
X              return rv;
X         }
X         return NULL;
X    case K_STRUCT:
X         return decl_su(0);
X    case K_UNION:
X         return decl_su(1);
X    case K_ENUM:
X         return decl_enum();
X    }
X
X    /* look for modifiers 'long', 'short', 'unsigned' */
```

```
X    adj = 0;
X    while (is_tadj(cur->e_token)) {
X         switch (cur->e_token) {
X         case K_SHORT:
X              adj |= SAW_SHORT;
X              break;
X         case K_LONG:
X              adj |= SAW_LONG;
X              break;
X         case K_UNSIGNED:
X              adj |= SAW_UNS;
X              break;
X         }
X         fadvnode();
X    }
X
X    /* look for base type 'char', 'int', 'float', 'double', 'void'*/
X    if (is_btype(cur->e_token)) {
X         btype = cur->e_token;
X         fadvnode();
X    } else if (adj == 0)    /* saw nothing */
X         return NULL;
X    else
X         btype = K_INT;
X
X    if (adj)
X         btype = adj_type(btype, adj);
X    rv = bas_type(btype);
X    return rv;
X}
X
XNODEP
Xdecl_enum()
X{
X    NODEP bas_type();
X
X    fadvnode();   /* skip 'enum' */
X
X    if (cur->e_token == ID) {   /* ignore tag */
X         fadvnode();
X    }
X    if (cur->e_token == '{') {  /* new declaration */
X         fadvnode();   /* skip '{' */
X         en_decls();   /* global scope */
X         if (cur->e_token != '}')
X              error("expect }");
X         else
X              fadvnode();   /* skip '}' */
X    }
X    return bas_type(K_INT);
X}
X
XNODEP
```

```
Xdecl_su(isunion)
X{
X    register NODEP rv, tagp;
X    NODEP *attab;
X    extern lineno;
X
X    fadvnode();   /* skip 'struct' or 'union' */
X
X    attab = level ? &blktab->b_tags : &tagtab;
X    tagp = NULL;
X    if (cur->e_token == ID) {   /* hold on to ID node */
X         tagp = cur;
X         e_to_t(tagp);
X         advnode();
X         nnmadd(tagp, isunion ? ".U" : ".S");
X    }
X    if (cur->e_token == '{') {  /* new declaration */
X         if (tagp == NULL) {    /* make fake name */
X              tagp = allocnode();
X              sprintf(tagp->n_name, isunion ? "%dU" :
X                        "%dS", lineno);
X         }
X         fadvnode();   /* skip '{' */
X         if (rv = llook(*attab, tagp)) {
X              freenode(tagp);
X              if (rv->n_right) {
X                   errors("struct redefined", rv->n_name);
X                   freenode(rv->n_right);
X                   rv->n_right = NULL;
X              }
X         } else {      /* new defn */
X              rv = tagp;
X              rv->t_token = isunion ? K_UNION : K_STRUCT;
X              rv->n_flags |= N_BRKPR; /* break print loops */
X              putlist(attab, rv);
X         }
X         su_decls(&rv->n_right, isunion,
X                   &rv->t_size, &rv->t_aln);
X         if (cur->e_token != '}')
X              error("expect }");
X         else
X              fadvnode();   /* skip '}' */
X    } else {      /* reference to old */
X         if (tagp == NULL) {
X              error("nonsense struct");
X              goto out;
X         }
X         /* ANSI special decl
X              struct <tag> ;
X            for hiding old tag within block */
X         if (cur->e_token == ';' && level)
X              rv = llook(*attab, tagp);
X         else
```

```
X             rv = alltags(tagp);
X         if (rv == NULL) {  /* delayed tag */
X             rv = tagp;
X             rv->t_token = isunion ? K_UNION : K_STRUCT;
X             rv->n_flags |= N_BRKPR; /* break print loops */
X             putlist(attab, rv);
X             goto out;
X         } else
X             freenode(tagp);
X    }
Xout:
X    return rv;
X}
X
XNODE *
Xalltags(np)
XNODE *np;
X{
X    register NODE *bp;
X    NODE *rv;
X
X    for (bp=blktab; bp != NULL; bp = bp->n_next)
X         if ((rv = llook(bp->b_tags, np)) != NULL)
X             return rv;
X    return llook(tagtab, np);
X}
X
XNODE *
Xallsyms(np)
XNODE *np;
X{
X    register NODE *bp;
X    NODE *rv;
X
X    for (bp=blktab; bp != NULL; bp = bp->n_next)
X         if ((rv = llook(bp->b_syms, np)) != NULL)
X             return rv;
X    return hlook(symtab, np);
X}
X
Xsim_type(a,b)
Xregister NODE *a, *b;
X{
Xmore:
X    if (a == b)
X         return 1;
X    if (a == NULL || b == NULL)
X         return 0;
X    if (a->t_token != b->t_token)
X         return 0;
X    if (a->t_size != b->t_size && a->t_size && b->t_size)
X         return 0;
X    a = a->n_tptr;
```

```
X    b = b->n_tptr;
X    goto more;
X}
X
X/* 2nd def of same name at same level */
X/* OK if one extern and types the same */
Xdef2nd(old,new)
XNODEP old, new;
X{
X    int osc, nsc;
X
X    if (sim_type(old->n_tptr, new->n_tptr) == 0)
X         goto bad;
X    osc = old->e_sc;
X    nsc = new->e_sc;
X    if (nsc == K_EXTERN) {  /* works only if no further use allowed */
X         freenode(new);
X         return 0;
X    }
X    if (osc == K_EXTERN) {
X         /* replace old def with new one */
X         /* for now, just put new one on list too */
X         return 1;
X    }
Xbad:
X    errorn("bad 2nd decl of ", new);
X    /* use 2nd def so other stuff works */
X    return 1;
X}
X
X/* saw fun but no body */
Xfix_fun(np)
XNODE *np;
X{
X    if (np == NULL) return;
X    if (np->n_tptr->t_token == '(') {     /* fix to extern */
X         if (np->e_sc != K_TYPEDEF)
X              np->e_sc = K_EXTERN;
X    }
X}
X
Xe_to_t(np)
XNODE *np;
X{
X    int token;
X
X    token = np->e_token;
X    np->t_token = token;
X    np->t_size = 0;
X    np->t_aln = 0;
X}
END_OF_FILE
if test 10944 -ne `wc -c <'hcc/DECL.C'`; then
```

```
    echo shar: \"'hcc/DECL.C'\" unpacked with wrong size!
fi
# end of 'hcc/DECL.C'
fi
if test -f 'hcc/EXPR.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/EXPR.C'\"
else
echo shar: Extracting \"'hcc/EXPR.C'\" \(9558 characters\)
sed "s/^X//" >'hcc/EXPR.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X * 
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  expr.c
X *
X *  Expression parse routines
X *
X *  All routines return either NULL or a valid tree
X *       binop nodes have non-null left and right
X *       unop nodes have non-null left
X *
X *  Special nodes:
X *       '(' : function call.  left:name-expr right:args
X *       ',': if part of function arg list, ival:num. descendants
X *       '?' : ?switch.  left:test-expr right:':' part
X *         ':' : left:true-expr right:false-expr
X *       TCONV: left:convertee tptr:type-list
X *       TSIZEOF: tptr:type-list
X *
X */
X
X#include <stdio.h>
X#include "param.h"
X#include "nodes.h"
X#include "tok.h"
X
Xextern NODEP cur;
XNODEP getnode();
XNODEP opt_follow();
X
Xextern int oflags[];
X#define debug oflags['x'-'a']
X
Xadvnode()
X{
X    cur = getnode();
X}
```

```
X
Xfadvnode()
X{
X    freenode(cur);
X    cur = getnode();
X}
X
XNODEP
Xgete_or_ty()
X{
X    NODEP getexpr(), makecast();
X    NODEP rv;
X
X    if (is_ty_start()) {
X         rv = makecast();
X         if (debug) {
X              printf("TY_X");
X              printnode(rv);
X         }
X         return rv;
X    } else
X         return getexpr();
X}
X
X/* call this for any expr including comma's */
XNODEP
Xgetexpr()
X{
X    NODEP np, get_f_expr();
X
X    np = get_f_expr(0);
X    return np;
X}
X
XNODEP
Xget_f_expr(flg)
Xint flg;
X{
X    NODEP assignx();
X    register NODEP op, lpart, rpart;
X    int i = 0;
X
X    lpart = assignx();
X    if (lpart == NULL) {
X         return NULL;
X    }
X    i++;
Xmore:
X    if (cur->e_token != ',')
X         return lpart;
X
X    op = cur;  advnode();
X    rpart = assignx();
```

```
X    if (rpart == NULL) {
X         error("',' expr syntax");
X         return lpart;
X    }
X    i++;
X    op->n_left = lpart;
X    op->n_right = rpart;
X    op->e_type = E_BIN;
X    op->e_ival = flg ? i : 0;
X    lpart = op;
X    if (debug) {
X         printf("COMMA");
X         printnode(op);
X    }
X    goto more;
X}
X
X/* call this if you want expr w/o comma's */
XNODEP
Xassignx()
X{
X    NODEP questx();
X    register NODEP op, lpart, rpart;
X
X    lpart = questx();
X    if (lpart == NULL)
X         return NULL;
X    if (!isassign(cur->e_token) && cur->e_token != '=')
X         return lpart;
X    op = cur;  advnode();
X    rpart = assignx();
X    if (rpart == NULL) {
X         error("'=op' expr syntax");
X         return lpart;
X    }
X    op->n_left = lpart;
X    op->n_right = rpart;
X    op->e_type = E_BIN;
X    if (debug) {
X         printf("ASSIGN");
X         printnode(op);
X    }
X    return op;
X}
X
X/* call this if you want expr w/o assign's or comma's */
X/* i.e. constant-expression */
XNODEP
Xquestx()
X{
X    NODEP binary();
X    register NODEP holdq, holdc;
X    NODEP qpart, tpart, fpart;
```

```
X
X    qpart = binary();
X    if (qpart == NULL)
X         return NULL;
X    if (cur->e_token != '?')
X         return qpart;
X    holdq = cur;  advnode();
X    tpart = questx();
X    if (tpart == NULL || cur->e_token != ':') {
Xbad:
X         error("'?:' expr syntax");
X         return qpart;
X    }
X    holdc = cur;  advnode();
X    fpart = questx();
X    if (fpart == NULL) goto bad;
X    holdc->n_left = tpart;
X    holdc->n_right = fpart;
X    holdc->e_type = E_BIN;
X    holdq->n_left = qpart;
X    holdq->n_right = holdc;
X    holdq->e_type = E_BIN;
X    if (debug) {
X         printf("QUEST");
X         printnode(holdq);
X    }
X    return holdq;
X}
X
XNODEP
Xbinary()
X{
X    NODEP unary(), buildbin();
X    register NODEP rv, op, e2;
X
X    rv = unary();
X    if (rv == NULL)
X         return NULL;
X    rv->e_prec = 0;
Xmore:
X    if (cur->e_prec == 0)   /* not binary op */
X         return rv;
X    op = cur;  advnode();
X    e2 = unary();
X    if (e2 == NULL) {
X         error("bin-op expr syntax");
X         return rv;
X    }
X    e2->e_prec = 0;
X    rv = buildbin(rv, op, e2);
X    if (debug) {
X         printf("BINARY");
X         printnode(rv);
```

```
X    }
X    goto more;
X}
X
XNODEP
Xbuildbin(lpart, op, upart)
XNODEP lpart, op, upart;
X{
X    register NODEP look, tail;
X    NODEP rv;
X
X    tail = NULL;
X    look = lpart;
X    for (look=lpart; op->e_prec < look->e_prec; look=look->n_right)
X         tail = look;
X    if (tail == NULL) {
X         op->n_left = lpart;
X         op->n_right = upart;
X         rv = op;
X    } else {
X         tail->n_right = op;
X         op->n_left = look;
X         op->n_right = upart;
X         rv = lpart;
X    }
X    op->e_type = E_BIN;
X    return rv;
X}
X
XNODEP
Xunary()
X{
X    register NODEP tp,e1;
X    NODEP primary();
X
X    if (cur->e_flags & CAN_U) {
X         tp = cur;  advnode();
X         if (tp->e_prec) {  /* also have BINARY op */
X              tp->e_token = UNARY tp->e_token;
X              strcat(tp->n_name, "U");
X         }
X         tp->n_left = unary();
X         tp->e_type = E_UNARY;
X         goto check;
X    } else
X    switch (cur->e_token) {
X    case '(':
X         fadvnode();
X         tp = gete_or_ty();
X         if (cur->e_token != ')') {
X              error("missing )");
X         } else
X              fadvnode();
```

```
X         if (tp == NULL)
X              return NULL;
X         if (tp->e_token == TCONV && tp->n_left == NULL) {
X              sprintf(tp->n_name, "cast to");
X              tp->n_left = unary();
X              tp->e_type = E_UNARY;
X         } else {
X              tp = opt_follow(tp);
X              goto out;
X         }
X         goto check;
X    case K_SIZEOF:
X         tp = cur;
X         advnode();
X         if (cur->e_token == '(') { /* may be type expr */
X              fadvnode();
X              e1 = gete_or_ty();
X              if (cur->e_token != ')') {
X                   error("missing )");
X              } else
X                   fadvnode();
X         } else
X              e1 = unary();
X         if (e1 == NULL) {
X              error("sizeof expr syntax");
X              return NULL;
X         }
X         if (e1->e_token == TCONV) {
X              freeunit(tp);
X              e1->e_token = TSIZEOF;
X              sprintf(e1->n_name, "T-sizeof");
X              tp = e1;
X              tp->e_type = E_LEAF;
X              goto out;
X         } else {
X              tp->e_type = E_UNARY;
X              tp->n_left = e1;
X         }
X         goto check;
X    default:
X         tp = primary();
X         goto out;
X    }
Xcheck:
X    if (tp == NULL) return NULL;
X    if (tp->n_left == NULL) {
X         error("u-op expr syntax");
X         return NULL;
X    }
Xout:
X    if (debug) {
X         printf("UNARY");
X         printnode(tp);
```

```
X    }
X    return tp;
X}
X
XNODEP
Xprimary()
X{
X    register NODEP e1;
X
X    switch (cur->e_token) {
X    case ID:
X    case ICON:
X    case FCON:
X    case SCON:
X         e1 = cur;
X         e1->e_type = E_LEAF;
X         advnode();
X         break;
X    case '(':
X         fadvnode();
X         e1 = getexpr();
X         if (cur->e_token != ')')
X              error("missing )");
X         else
X              fadvnode();
X         break;
X    default:
X         e1 = NULL;
X    }
X    if (e1 == NULL)
X         return NULL;
X    return opt_follow(e1);
X}
X
XNODEP
Xopt_follow(np)
XNODEP np;
X{
X    register NODEP tp, e1, t2;
X
X    switch (cur->e_token) {
X    case '[':
X         tp = cur;  advnode();
X         e1 = getexpr();
X         if (cur->e_token != ']') {
X              error("missing ]");
X              return np;
X         } else {
X              t2 = cur;  advnode();
X         }
X         if (e1 == NULL) {
X              error("empty []");
X              return np;
```

```
X        }
X        t2->n_left = np;
X        t2->n_right = e1;
X        t2->e_type = E_BIN;
X        t2->e_token = '+';
X        strcpy(t2->n_name, "+ [");
X
X        tp->n_left = t2;
X        tp->e_type = E_UNARY;
X        tp->e_token = STAR;
X        strcpy(tp->n_name, "U*");
X
X        goto out;
X    case '(':
X        tp = cur;
X        advnode();
X        e1 = get_f_expr(1);
X        if (cur->e_token != ')')
X            error("expect )");
X        else
X            fadvnode();
X        tp->n_left = np;
X        tp->n_right = e1;
X        tp->e_type = E_SPEC;
X        goto out;
X    case '.':
X    case ARROW:
X        tp = cur;  advnode();
X        if (cur->e_token != ID) {
X            error("expect ID");
X            return np;
X        }
X        tp->n_left = np;
X        tp->n_right = cur;
X        tp->e_type = E_SPEC;
X        if (tp->e_token == ARROW) { /* make into (*X).Y */
X            tp->e_token = '.';
X            strcpy(tp->n_name, ".");
X
X            t2 = allocnode();
X            t2->e_token = STAR;
X            t2->n_left = np;
X            t2->e_type = E_UNARY;
X            strcpy(t2->n_name, "U*");
X
X            tp->n_left = t2;
X        }
X        advnode();
X        goto out;
X    case DOUBLE '+':
X    case DOUBLE '-':
X        tp = cur;  advnode();
X        tp->e_token = (tp->e_token == DOUBLE '+') ? POSTINC : POSTDEC;
```

```
X         strcat(tp->n_name, "post");
X         tp->n_left = np;
X         tp->e_type = E_UNARY;
X         goto out;
X    default:
X         return np;
X    }
Xout:
X    return opt_follow(tp);
X}
X
X/* restricted version of unary for declarations or coertions */
X/* allows NULL primary part */
XNODEP
Xdeclarator()
X{
X    register NODEP tp,e1;
X    NODEP ty_primary(), ty_follow();
X
X    if (cur->e_token == '*') {
X         tp = cur;
X         tp->e_token = UNARY tp->e_token;
X         strcat(tp->n_name, "U");
X         advnode();
X         tp->n_left = declarator();
X         return tp;
X    } else
X    switch (cur->e_token) {
X    case '(':
X         tp = cur;
X         advnode();
X         e1 = declarator();
X         if (cur->e_token != ')') {
X              error("expect )");
X         } else
X              fadvnode();
X         if (e1 == NULL) {  /* special "fun of" */
X              /* left and right already NULL */
X              return ty_follow(tp);
X         } else {
X              freeunit(tp);
X              return ty_follow(e1);
X         }
X    default:
X         return ty_primary();
X    }
X}
X
X/* restricted version of primary for "declarator" */
XNODEP
Xty_primary()
X{
X    register NODEP e1;
```

```
X    NODEP ty_follow();
X
X    switch (cur->e_token) {
X    case ID:
X         e1 = cur;
X         advnode();
X         break;
X    case '(':
X         fadvnode();
X         e1 = declarator();
X         if (cur->e_token != ')')
X              error("expect )");
X         else
X              fadvnode();
X         break;
X    default:
X         e1 = NULL;
X    }
X    return ty_follow(e1);
X}
X
X/* restricted version of opt_follow for 'declarator' */
X/* allow null [] */
XNODEP
Xty_follow(np)
XNODEP np;
X{
X    register NODEP tp, e1;
X    NODEP ty_args();
X
X    switch (cur->e_token) {
X    case '[':
X         tp = cur;
X         advnode();
X         e1 = questx();
X         if (cur->e_token != ']')
X              error("expect ]");
X         else
X              fadvnode();
X         tp->n_left = np;
X         tp->n_right = e1;
X         goto out;
X    case '(':
X         tp = cur;
X         advnode();
X         e1 = ty_args();    /* allow args of fun to follow */
X         if (cur->e_token != ')')
X              error("expect )");
X         else
X              fadvnode();
X         tp->n_left = np;
X         tp->n_right = e1;
X         goto out;
```

```
X    default:
X         return np;
X    }
Xout:
X    return ty_follow(tp);
X}
X
X/* called for args of function declaration or NULL */
XNODEP
Xty_args()
X{
X    NODEP opt_id();
X    register NODEP rv, tail, new;
X
X    rv = opt_id();
X    if (rv == NULL)
X         return NULL;
X    tail = rv;
Xmore:
X    if (cur->e_token != ',')
X         return rv;
X    fadvnode();
X    new = opt_id();
X    if (new == NULL) {
X         error("expect as-op value");
X         return rv;
X    }
X    tail->n_left = new;
X    tail = new;
X    goto more;
X}
X
XNODEP
Xopt_id()
X{
X    NODEP rv;
X
X    if (cur->e_token == ID) {
X         rv = cur;
X         advnode();
X         return rv;
X    } else
X         return NULL;
X}
END_OF_FILE
if test 9558 -ne `wc -c <'hcc/EXPR.C'`; then
    echo shar: \"'hcc/EXPR.C'\" unpacked with wrong size!
fi
# end of 'hcc/EXPR.C'
fi
if test -f 'hcc/TOK.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/TOK.C'\"
else
```

```
echo shar: Extracting \"'hcc/TOK.C'\" \(10934 characters\)
sed "s/^X//" >'hcc/TOK.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  tok.c
X *
X *  Basic level token routines
X *
X *  At this level, we return the following things:
X *       id's - strings of alpha-alnum
X *       integer constants
X *       float constants
X *       string constants
X *       multi-char tokens
X *
X *  We DONT know about:
X *       keywords
X *       #defined id's
X *       any other meaning of a name
X *
X *  Interface:
X *       call nxttok() to get next token
X *       look at 'curtok' for current token
X *       note that curtok.name points to a static area
X *         for ID or SCON
X *
X *       if EOF is seen, we call endfile() before
X *         giving up
X *
X *  Special flags:  (tk_flags)
X *       These special flags are needed for the pre-processor.
X *       All but TK_SEENL are 1-shot.
X *
X *       TK_SEENL - want to see \n
X *       TK_WS - want to see white space (for #define)
X *       TK_NOESC - dont do '\' escapes in strings
X *       TK_LTSTR - '<' is a string starter
X *       TK_ONLY1 - skip to token after \n  (for #if--)
X */
X
X#include <stdio.h>
X#include "param.h"
X#include "tok.h"
X
X#if dLibs
```

```
X#include <ctype.h>
X#endif
X
Xstruct tok curtok;
Xchar curstr[MAXSTR+1];
X
X#define TK_SEENL  1    /* want to see NL token */
X#define TK_SEEWS  2    /* want to see WS token */
X#define TK_ONLY1  4    /* only want 1st token on line */
X#define TK_LTSTR  8    /* '<' starts a string */
X#define TK_NOESC  16   /* dont do '\' escapes in string */
X
Xint tk_flags, sawnl;
X
Xextern FILE *input;
Xextern int lineno;
X
X#define NOCHAR    0x100
X
X#ifdef DEBUG
Xextern int oflags[];
X#define debug oflags['b'-'a']
X#endif
X
Xnxttok()
X{
X    register struct tok *t;
X    char *getname();
X    long getnum();
X    register int c;
X    double getfrac();
X
X    t = &curtok;
X    t->name = curstr;
X    t->name[0] = 0;
X    t->prec = 0;
X    t->flags = 0;
Xmore:
X    c = mygetchar();
X    if (c == EOF) {
X         tk_flags = 0;
X         return 0;
X    }
X    if (c == '\n') {
X         tk_flags &= ~TK_ONLY1;
X         if ((tk_flags & TK_SEENL) == 0)
X              goto more;
X         t->tnum = NL;
X         t->name = "\n";
X         goto out;
X    }
X    if (tk_flags & TK_ONLY1)
X         goto more;
```

```
X    if (c <= ' ') {
X         if ((tk_flags & TK_SEEWS) == 0)
X              goto more;
X         t->tnum = WS;
X         t->name = " ";
X         goto out;
X    }
X    if (c >= '0' && c <= '9') {
X         t->tnum = ICON;
X         t->ival = getnum(c);
X         if (lookfrac(t->ival) || lookexp(t->ival,0.0))
X              goto out;
X    moresuf:
X         c = mygetchar();
X         if (tolower(c) == 'l') {
X              t->flags |= SEE_L;
X              goto moresuf;
X         } else if (tolower(c) == 'u') {
X              t->flags |= SEE_U;
X              goto moresuf;
X         } else {
X              myungetc(c);
X         }
X         sprintf(curstr, "%ld",
X              t->ival);
X         goto out;
X    }
X    if (isalpha(c) || c == '_') {
X         t->tnum = ID;
X         t->name = getname(c);
X         goto out;
X    }
X    if (c == '.') {
X         c = mygetchar();
X         if (c >= '0' && c <= '9') {
X              gotfrac(0L, getfrac(c));
X              goto out;
X         } else {
X              myungetc(c);
X              matchop('.');
X              goto out;
X         }
X    }
X    if(matchop(c) == 0)
X         goto more;
Xout:
X    if (debug) printf("<%s>", t->name);
X    tk_flags &= TK_SEENL;  /* all but SEENL are 1-shot */
X    return 1;
X}
X
Xlong
Xgetnum(c)
```

```
Xregister int c;
X{
X    register long val = 0;
X    int base, i;
X
X    if (c == '0') {
X         base = 8;
X    } else {
X         base = 10;
X         val = c - '0';
X    }
Xmore:
X    c = mygetchar();
X    if (c == EOF)
X         return val;
X    if (tolower(c) == 'x' && val == 0) {
X         base = 16;
X         goto more;
X    }
X    if (c >= '0' && c <= '9') {
X         val = base*val + (c - '0');
X         goto more;
X    }
X    if (base == 16 && (i = ishexa(c))) {
X         val = 16*val + i;
X         goto more;
X    }
X    myungetc(c);
X    return val;
X}
X
Xdouble
Xgetfrac(c)
Xregister c;
X{
X    register double val;
X    register double dig = 0.1;
X
X    val = dig * (c - '0');
Xmore:
X    c = mygetchar();
X    if (c >= '0' && c <= '9') {
X         dig = .1 * dig;
X         val += dig * (c - '0');
X         goto more;
X    }
X    myungetc(c);
X    return val;
X}
X
Xlookfrac(intpart)
Xlong intpart;
X{
```

```
X    int c;
X    double frac;
X
X    c = mygetchar();
X    if (c != '.') {
X         myungetc(c);
X         return 0;
X    }
X    c = mygetchar();
X    if (c >= '0' && c <= '9') {
X         frac = getfrac(c);
X    } else {
X         myungetc(c);
X         frac = 0.0;
X    }
X    gotfrac(intpart, frac);
X    return 1;
X}
X
Xgotfrac(intpart, frac)
Xlong intpart;
Xdouble frac;
X{
X    if (lookexp(intpart, frac) == 0)
X         makeflt(intpart, frac, 0);
X}
X
Xlookexp(intpart, frac)
Xlong intpart;
Xdouble frac;
X{
X    int c;
X    int minus;
X    int exp;
X
X    minus = 0;
X    c = mygetchar();
X    if (tolower(c) != 'e') {
X         myungetc(c);
X         return 0;
X    }
X    c = mygetchar();
X    if (c == '-') {
X         minus = 1;
X         c = mygetchar();
X    } else if (c == '+')
X         c = mygetchar();
X    if (c >= '0' && c <= '9') {
X         exp = getnum(c);
X    } else {
X         exp = 0;
X         myungetc(c);
X    }
```

```
X    if (minus)
X         exp = -exp;
X    makeflt(intpart, frac, exp);
X    return 1;
X}
X
Xmakeflt(intpart, frac, exp)
Xlong intpart;
Xdouble frac;
X{
X    register double val;
X    double mod, mod10;
X    register struct tok *t;
X
X    val = intpart + frac;
X    if (exp > 0) {
X         mod = 1e1;
X         mod10 = 1e10;
X    } else if (exp < 0) {
X         mod = 1e-1;
X         mod10 = 1e-10;
X         exp = -exp;
X    }
X    while (exp >= 10) {
X         val *= mod10;
X         exp -= 10;
X    }
X    while (exp--)
X         val *= mod;        /* slow and dirty */
X    t = &curtok;
X    t->tnum = FCON;
X    t->fval = val;
X    sprintf(t->name, FLTFORM, val);
X}
X
Xchar *
Xgetname(c)
Xregister int c;
X{
X    register int nhave;
X
X    nhave = 0;
X    do {
X         if (nhave < MAXSTR)
X              curstr[nhave++] = c;
X         c = mygetchar();
X    } while (isalnum(c) || c == '_');
X    myungetc(c);
X    curstr[nhave] = 0;
X    return curstr;
X}
X
Xstatic char *holdstr;
```

```
X
Xchr_push(s)
Xchar *s;
X{
X    holdstr = s;
X}
X
Xstatic int holdchar, xholdchar;
X
Xmygetchar()
X{
X    register int c;
X    int c2;
X
X    if (holdchar) {
X         c = holdchar;
X         holdchar = 0;
X         goto out;
X    }
X    if (holdstr) {          /* used for -D args */
X         c = *holdstr++;
X         if (c == 0) {
X              holdstr = NULL;
X              return '\n';
X         }
X         return c;
X    }
X
Xretry:
X    c = xgetc();
X    if (c == EOF) {
X         if (endfile())
X              goto retry;
X    } else if (c == '\\') {     /* ansi handling of backslash nl */
X         c2 = xgetc();
X         if (c2 == '\n') {
X              lineno++;
X              goto retry;
X         } else
X              xholdchar = c2;
X    }
Xout:
X    if (c == '\n') {
X         sawnl++; /* for pre.c */
X         lineno++;
X    }
X    return c;
X}
X
Xxgetc()
X{
X    register int c;
X
```

```
X    if (xholdchar) {
X         c = xholdchar;
X         xholdchar = 0;
X         return c;
X    }
X#if CC68|dLibs
X    if (input == stdin)    /* bypass stupid input */
X         c = hackgetc();
X    else
X#endif
X         c = getc(input);
X    if (c != EOF)
X         c &= 0x7f;
X    return c;
X}
X
Xmyungetc(c)
Xchar c;
X{
X    if (c != EOF)
X         holdchar = c;
X    if (c == '\n')
X         lineno--;
X}
X
Xstruct op {
X    char *name;
X    char *asname;
X    int flags;
X    char prec;
X    char value;
X} ops[] = {
X    {"{"},
X    {"}"},
X    {"["},
X    {"]"},
X    {"("},
X    {")"},
X    {"#"},
X    {"\\"},
X    {";"},
X    {","},
X    {":"},
X    {"."},
X
X    {"\"", 0, SPECIAL},
X    {"'", 0, SPECIAL},
X
X    {"==", 0, C_NOT_A, 5},
X    {"=", 0, 0},
X
X    {"++", 0, CAN_U},
X    {"+", "+=", CAN_AS|C_AND_A, 2},
```

```
X
X    {"--", 0, CAN_U},
X    {"->", 0, 0, 0, ARROW},
X    {"-", "-=", CAN_U|CAN_AS, 2},
X
X    {"*", "*=", CAN_U|CAN_AS|C_AND_A, 1},
X    {"%", "%=", CAN_AS, 1},
X
X    {"/*", 0, SPECIAL},
X    {"/", "/=", CAN_AS, 1},
X
X    {"&&", 0, 0, 9},
X    {"&", "&=", CAN_U|CAN_AS|C_AND_A, 6},
X
X    {"||", 0, 0, 10},
X    {"|", "|=", CAN_AS|C_AND_A, 8},
X
X    {"!=", 0, C_NOT_A, 5, NOTEQ},
X    {"!", 0, CAN_U},
X
X    {"~", 0, CAN_U},
X
X    {"^", "^=", CAN_AS|C_AND_A, 7},
X
X    {"<<", "<<=", CAN_AS, 3},
X    {"<=", 0, C_NOT_A, 4, LTEQ},
X    {"<", 0, SPECIAL|C_NOT_A, 4},
X
X    {">>", ">>=", CAN_AS, 3},
X    {">=", 0, C_NOT_A, 4, GTEQ},
X    {">", 0, C_NOT_A, 4},
X
X    {"?", 0, 0},
X
X    {0, 0, 0}
X};
X
X#define FIRST_C   '!'
X#define LAST_C    0177
Xstruct op *opstart[LAST_C-FIRST_C+1];
X
Xmo_init()
X{
X    register struct op *p;
X    register c;
X
X    for (p=ops; p->name; p++) {
X         c = p->name[0];
X         if (opstart[c-FIRST_C] == 0)
X              opstart[c-FIRST_C] = p;
X    }
X}
X
```

```
Xmatchop(c)
X{
X    register struct tok *t;
X    register struct op *p;
X    int nxt;
X    int value;
X    static first = 0;
X
X    t = &curtok;
X    nxt = mygetchar();
X    value = c;
X    if (first == 0) {
X         mo_init();
X         first = 1;
X    }
X    p = opstart[c-FIRST_C];
X    if (p)
X    for (; p->name; p++)
X         if (p->name[0] == c)
X              if (p->name[1] == 0 || p->name[1] == nxt) {
X                   if (p->name[1] == 0)
X                        myungetc(nxt);
X                   else {
X                        value = p->value ? p->value :
X                             DOUBLE value;
X                   }
X                   if (p->flags & SPECIAL)
X                        if (c != '<' ||
X                          tk_flags & TK_LTSTR)
X                             return dospec(p);
X                   t->flags = p->flags;
X                   if (p->flags & CAN_AS) {
X                        nxt = mygetchar();
X                        if (nxt != '=') {
X                             myungetc(nxt);
X                        } else {
X                             value = ASSIGN value;
X                             t->flags = 0;
X                        }
X                   }
X                   t->name = isassign(value)?p->asname:p->name;
X                   t->tnum = value;
X                   t->prec = isassign(value)? 0 : p->prec;
X                   return 1;
X              }
X    myungetc(nxt);
X    t->name = "???";
X    t->tnum = BADTOK;
X    return 0;
X}
X
Xdospec(p)
Xstruct op *p;
```

```
X{
X    register struct tok *t;
X    register int c;
X    int nhave;
X    int endc;
X
X    t = &curtok;
X    switch (p->name[0]) {
X    case '/':     /* slash-star */
Xlook:
X         do {
X              c = mygetchar();
X         } while (c != '*');
X         c = mygetchar();
X         if (c == '/')
X              return 0;
X         myungetc(c);
X         goto look;
X    case '\'':
X         t->tnum = ICON;
X         t->ival = getschar('\''); /* allow only 1 for now*/
X         while (getschar('\'') != NOCHAR)
X              ;
X         sprintf(curstr, "%d", (int)t->ival);
X         return 1;
X    case '<':
X         endc = '>';
X         t->tnum = SCON2;
X         goto strs;
X    case '"':
X         endc = '"';
X         t->tnum = SCON;
X    strs:
X         t->name = curstr;
X         nhave = 0;
X         c = getschar(endc);
X         while (c != NOCHAR) {
X              if (c >= 0 && c <= 1 && nhave < MAXSTR) {
X                   /* allow null */
X                   curstr[nhave++] = 1;
X                   c++;
X              }
X              if (nhave < MAXSTR)
X                   curstr[nhave++] = c;
X              c = getschar(endc);
X         }
X         curstr[nhave] = 0;
X         return 1;
X    }
X}
X
Xgetoct(c)
X{
```

```
X    char n, i;
X
X    n = c - '0';
X    for (i=1; i < 3; i++) {
X         c = mygetchar();
X         if (c < '0' || c > '7') {
X              myungetc(c);
X              return (int)n;
X         }
X         n = 8*n + (c - '0');
X    }
X    return (int)n;
X}
X
Xgetschar(del)
Xchar del;
X{
X    register int c;
X
Xmore:
X    c = mygetchar();
X    if (c == del)
X         return NOCHAR;
X    if (c == '\n') {
X         error("nl in string");
X         myungetc(c);
X         return NOCHAR;
X    }
X    if (c == '\\' && (tk_flags & TK_NOESC) == 0) {
X         c = mygetchar();
X         if (c == del)
X              return c;
X         if (c >= '0' && c <= '7')
X              return getoct(c);
X         switch (c) {
X/*
X         case '\n':
X              goto more;
X*/
X         case 'b':
X              c = '\b';
X              break;
X         case 'n':
X              c = '\n';
X              break;
X         case 't':
X              c = '\t';
X              break;
X         case 'r':
X              c = '\r';
X              break;
X         case 'f':
X              c = '\f';
```

```
X             break;
X         }
X    }
X    return c;
X}
X
X#if !dLibs
X
Xisalpha(c)
Xregister char c;
X{
X    if ((c>='a' && c<='z') ||
X         (c>='A' && c<='Z'))
X         return 1;
X    return 0;
X}
X
Xisalnum(c)
Xregister char c;
X{
X    return (isalpha(c) || (c>='0' && c<='9'));
X}
X
Xtolower(c)
Xregister char c;
X{
X    if (c>='A' && c<='Z')
X         c += 'a'-'A';
X    return c;
X}
X
X#endif
X
Xishexa(c)
Xregister char c;
X{
X    if (c>='a' && c<='f')
X         return (c-'a'+10);
X    if (c>='A' && c<='F')
X         return (c-'A'+10);
X    return 0;
X}
X
X#if CC68
Xhackgetc()
X{
X    register int c;
X
X    c = bios(2,2);
X    switch (c) {
X    case 4:
X         return EOF;
X    case '\r':
```

```
X    case '\n':
X         bios(3,2,'\r');
X         bios(3,2,'\n');
X         return '\n';
X    }
X    bios(3,2,c);
X    return c;
X}
X#endif
X
X#if dLibs
Xhackgetc()
X{
X    register int c;
X
X    c = getchar();
X    switch (c) {
X    case 4:
X         return EOF;
X    case '\n':
X         putchar('\n');
X         break;
X    }
X    return c;
X}
X#endif
END_OF_FILE
if test 10934 -ne `wc -c <'hcc/TOK.C'`; then
    echo shar: \"'hcc/TOK.C'\" unpacked with wrong size!
fi
# end of 'hcc/TOK.C'
fi
echo shar: End of archive 4 \(of 9\).
cp /dev/null ark4isdone
MISSING=""
for I in 1 2 3 4 5 6 7 8 9 ; do
    if test ! -f ark${I}isdone ; then
     MISSING="${MISSING} ${I}"
    fi
done
if test "${MISSING}" = "" ; then
    echo You have unpacked all 9 archives.
    rm -f ark[1-9]isdone ark[1-9][0-9]isdone
else
    echo You still need to unpack the following archives:
    echo "        " ${MISSING}
fi
##  End of shell archive.
exit 0


From -5389932677213612110
X-Google-Language: ENGLISH,ASCII-7-bit
```

X-Google-Thread: 100493,fdf98513a8e34bf5,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-24 18:48:06 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!uwm.edu!gem.mps.ohio-state.edu!
apple!sun-barr!newstop!sun!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i096: sozobon1.2 -- Update to Sozobon C compiler part05/09
Keywords: shar, C
Message-ID: <34607@grapevine.uucp>
Date: 25 Oct 89 01:48:06 GMT
Sender: news@grapevine.EBay
Reply-To: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Lines: 2173
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Posting-number: Volume 2, Issue 96
Archive-name: sozobon1.2/part05

```
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of archive 5 (of 9)."
# Contents:  hcc/FUN.C hcc/G2.C hcc/GEN.C
# Wrapped by koreth@panarthea on Tue Oct 24 18:40:46 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'hcc/FUN.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/FUN.C'\"
else
echo shar: Extracting \"'hcc/FUN.C'\" \(11588 characters\)
sed "s/^X//" >'hcc/FUN.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  fun.c
X *
X *  Handle function entry, exit, etc.
X *  Parse statements.
X *  Also, general syntax error recovery strategy.
X */
X
X#include <stdio.h>
```

```
X#include "param.h"
X#include "tok.h"
X#include "nodes.h"
X#include "cookie.h"
X
X#if MMCC
Xoverlay "pass2"
X#endif
X
Xextern NODE *cur;
X
Xint level;
XNODE *blktab;
XNODE *labels;
X
Xstruct swittbl {
X    NODEP    caselist;
X    int  deflbl;
X} *curswit;
X
Xint curbrk, curcont;
Xint funtopl, funbotl, funretl, funstrl;
XNODEP funtyp;
Xint maxregs;
Xlong maxlocs;
X
Xint skipon;
X
XNODEP glb_decls();
X
Xextern int oflags[];
X#define debugl oflags['l'-'a']
X#define debugs oflags['s'-'a']
X#define debugv oflags['v'-'a']
X
Xfindtok(x)
X{
X    while (cur->e_token != EOFTOK && cur->e_token != x)
X        fadvnode();
X    if (cur->e_token == EOFTOK)
X        exit(1);
X}
X
Xprogram()
X{
X    extern NODEP symtab[];
X    NODEP last;
X
X    skipon = 0;
Xmore:
X    last = glb_decls();
X    if (cur->e_token == EOFTOK)
X        return;
```

```
X    if (last) skipon = 0;   /* saw something valid */
X    if (last && last->n_tptr && last->n_tptr->t_token == '(') {
X    /* possible function definition */
X         if (debugs) {
X              printf("FUN ");
X              put_nnm(last);
X         }
X         out_fstart(last);
X#ifdef OUT_AZ
X         last->e_sc = HERE_SC;
X#else
X         last->e_sc = K_EXTERN;
X#endif
X         fun_start(last->n_tptr);
X         args_blk(last->n_tptr->n_right);
X         sub_block();
X         fun_end();
X         clr_lvl(); /* for args block */
X         goto more;
X    }
X    /* error if get to here */
X    if (last) {
X         error("missing ;");
X         goto more;
X    } else {
X         skip();
X         goto more;
X    }
X}
X
Xfun_start(np)
XNODEP np;
X{
X    NODEP functy();
X
X    funtyp = functy(np);
X    curbrk = curcont = -1;
X    funtopl = new_lbl();
X    funbotl = new_lbl();
X    funretl = new_lbl();
X    switch (funtyp->t_token) {
X    case K_STRUCT:
X    case K_UNION:
X         funstrl = new_lbl();
X         break;
X    default:
X         funstrl = 0;
X    }
X    maxregs = 0;
X    maxlocs = 0;
X    out_br(funbotl);
X    def_lbl(funtopl);
X}
```

```
X
Xfun_end()
X{
X    NODEP np;
X
X    if (labels) {
X         for (np = labels; np; np = np->n_next)
X              if (np->c_defined == 0)
X                   errorn("undefined label", np);
X         freenode(labels);
X         labels = NULL;
X    }
X    def_lbl(funretl);
X    out_fret(maxregs, funstrl);
X    def_lbl(funbotl);
X    out_fend(maxregs, maxlocs);
X    out_br(funtopl);
X    if (funstrl)
X         out_fs(funstrl, funtyp->t_size);
X}
X
Xskip()
X{
X    if (skipon == 0) {
X         error("syntax (try skipping...)");
X         skipon = 1;
X    }
X    fadvnode();
X}
X
Xblock()
X{
X    int some;
X    int sawsome;
X
X    some = loc_decls();
X    if (cur->e_token == EOFTOK)
X         return;
X    if (some) skipon = 0;
Xmore:
X    sawsome = stmts();
X    if (sawsome) skipon = 0;
X    if (cur->e_token == '}') {
X         if (blktab->b_regs > maxregs)
X              maxregs = blktab->b_regs;
X         if (blktab->b_size + blktab->b_tsize > maxlocs)
X              maxlocs = blktab->b_size + blktab->b_tsize;
X         return;
X    }
X
X    /* error if get to here */
X    if (cur->e_token == EOFTOK || is_tykw(cur->e_token))
X         /* get out of block */
```

```
X         return;
X    else {
X         skip();
X         goto more;
X    }
X}
X
Xclr_lvl()
X{
X    NODE *bp;
X
X    level--;
X    bp = blktab;
X    blktab = bp->n_next;
X    bp->n_next = NULL;
X    if (debugl && bp->b_syms) {
X         printf("local syms %d", level);
X         printlist(bp->b_syms);
X    }
X#ifdef OUT_AZ
X    xrefl(bp->b_syms);
X#endif
X    freenode(bp->b_syms);
X    if (debugl && bp->b_tags) {
X         printf("local tags %d", level);
X         printlist(bp->b_tags);
X    }
X    freenode(bp->b_tags);
X    freenode(bp);
X}
X
Xeat(c)
X{
X    char *p = "assume X";
X
X    if (cur->e_token == c)
X         fadvnode();
X    else {
X         p[strlen(p) - 1] = c;
X         error(p);
X    }
X}
X
Xsub_block()
X{
X    register NODE *new;
X
X    if (debugs)
X         printf("{ ");
X    eat('{');
X    level++;
X    new = allocnode();
X    new->n_next = blktab;
```

```
X    sprintf(new->n_name, "sub{");
X    blktab = new;
X    block();
X    clr_lvl();
X    eat('}');
X    if (debugs)
X         printf("}\n");
X}
X
Xargs_blk(np)
XNODEP np;
X{
X    register NODE *p;
X    register NODE *new;
X    NODE *tp;
X    NODEP llook();
X    long size;
X    int rmask;
X
X    size = 0;
X    rmask = 0;
X    new = allocnode();
X    new->n_next = blktab;
X    sprintf(new->n_name, "arg{");
X    blktab = new;
X    level++;
X    loc_decls();
X    /* make sure all decls were in arg list */
X    for (p=new->b_syms; p != NULL; p = p->n_next)
X         if (llook(np, p) == NULL)
X              errorn("ID not param", p);
X    /* now make any names not mentioned INT */
X    /* and generate offsets and alloc regs */
X    for (p=np; p != NULL; p = p->n_next) {
X         if ((tp=llook(new->b_syms, p)) == NULL) {
X              def_arg(&new->b_syms, p);
X              tp = new->b_syms;
X         }
X         lc_size(&size, &rmask, tp);
X         if (tp->e_sc == K_REGISTER)
X              reg_arg(&rmask, tp);
X         if (debugv) {
X              printf("final o%ld r%d ", tp->e_offs, tp->e_rno);
X              put_nnm(tp);
X              putchar('\n');
X         }
X         out_advice(tp);
X    }
X    new->b_regs = rmask;
X}
X
Xreg_arg(rp, xp)
Xint *rp;
```

```
XNODEP xp;
X{
X    if (lc_reg(rp, xp) == 0) {  /* out of regs? */
X         xp->e_sc = K_AUTO;
X         return;
X    }
X    out_argreg(xp);
X}
X
X
Xstmts()
X{
X    int didsome;
X
X    didsome = 0;
X    while (stmt())
X         didsome++;
X    return didsome;
X}
X
Xstmt_bc(brk,cont)
X{
X    int svb, svc;
X
X    svb = curbrk;
X    svc = curcont;
X    curbrk = brk;
X    curcont = cont;
X
X    stmt();
X
X    curbrk = svb;
X    curcont = svc;
X}
X
Xstmt_b(brk)
X{
X    int svb;
X
X    svb = curbrk;
X    curbrk = brk;
X
X    stmt();
X
X    curbrk = svb;
X}
X
X/* do a single statement */
Xstmt()
X{
X    register tok;
X    NODEP np;
X    NODEP getexpr();
```

```
X    int i;
X
Xmore:
X    tok = cur->e_token;
X    if (is_stkw(tok)) {
X         if (is_blkst(tok)) {
X              i = blk_stmt();
X         } else if (is_brast(tok)) {
X              i = bra_stmt();
X         } else if (is_lblst(tok)) {
X              i = lbl_stmt();
X         } else {
X              asm_stmt();
X              return 1;
X         }
X         if (i == 0)
X              goto more;
X         return 1;
X    }
X    else if (tok == '{') {
X         sub_block();
X         return 1;
X    } else if (tok == ';') {
X         fadvnode();
X         return 1;
X    }
X    np = getexpr();
X    if (np) {
X         if (cur->e_token == ':') {
X              fadvnode();
X              label(np);
X              goto more;
X         }
X         expr_stmt(np);
X         if (cur->e_token != ';')
X              error("missing ;");
X         else
X              fadvnode();
X         return 1;
X    }
X    return 0;
X}
X
Xexpr_stmt(np)
XNODEP np;
X{
X    if (debugs) {
X         printf("E_STMT ");
X         if (debugs > 1)
X              printnode(np);
X    }
X    do_expr(np, FORSIDE);
X}
```

```
X
Xlabel(np)
XNODEP np;
X{
X    register NODEP tp;
X    NODEP llook();
X
X    if (debugs) {
X         printf("LABEL ");
X         if (debugs > 1)
X              printnode(np);
X    }
X    if (np->e_token != ID) {
X         error("weird label");
X         return;
X    }
X    tp = llook(labels, np);
X    if (tp) {
X         freenode(np);
X         if (tp->c_defined) {
X              error("duplicate label");
X              return;
X         }
X    } else {
X         putlist(&labels, np);
X         tp = np;
X         tp->c_casel = new_lbl();
X    }
X    tp->c_defined = 1;
X    def_lbl(tp->c_casel);
X}
X
Xblk_stmt()
X{
X    register tok;
X    int l1, l2, l3;
X    NODEP e1, e2, e3;
X    NODEP opt_expr(), paren_expr(), def_type();
X    struct swittbl locswit, *oldp;
X    extern int lineno;
X    int svline, svline2;
X
X    tok = cur->e_token;
X    fadvnode();
X    switch (tok) {
X    case K_IF:
X         if (debugs)
X              printf("IF ");
X         l1 = new_lbl();
X         e1 = paren_expr();
X         gen_brf(e1, l1);
X         eat(')');
X         stmt();
```

```
X         opt_else(l1);
X         return 1;
X    case K_WHILE:
X         if (debugs)
X             printf("WHILE ");
X         e1 = paren_expr();
X         l1 = new_lbl();
X         l2 = new_lbl();
X
X         def_lbl(l1);
X         gen_brf(e1,l2);
X         eat(')');
X
X         stmt_bc(l2,l1);
X
X         out_br(l1);
X         def_lbl(l2);
X         return 1;
X    case K_DO:
X         if (debugs)
X             printf("DO ");
X         l1 = new_lbl();
X         l2 = new_lbl();
X         l3 = new_lbl();
X         def_lbl(l1);
X
X         stmt_bc(l3,l2);
X
X         def_lbl(l2);
X         eat(K_WHILE);
X         e1 = paren_expr();
X         gen_brt(e1, l1);
X         eat(')');
X         eat(';');
X         def_lbl(l3);
X         return 1;
X    case K_FOR:
X         if (debugs)
X             printf("FOR ");
X         l1 = new_lbl();
X         l2 = new_lbl();
X         l3 = new_lbl();
X         eat('(');
X         e1 = opt_expr();
X         expr_stmt(e1);
X         eat(';');
X         def_lbl(l1);
X         e2 = opt_expr();
X         if (e2)
X             gen_brf(e2,l3);
X         eat(';');
X         e3 = opt_expr();   /* save for later */
X         svline = lineno;
```

```
X        eat(')');
X
X        stmt_bc(l3,l2);
X
X        def_lbl(l2);
X
X        svline2 = lineno;
X        lineno = svline;
X        expr_stmt(e3);
X        lineno = svline2;
X
X        out_br(l1);
X        def_lbl(l3);
X        return 1;
X    case K_SWITCH:
X        if (debugs)
X            printf("SWITCH ");
X        e1 = paren_expr();
X        l1 = new_lbl();
X        l2 = new_lbl();
X        to_d0(e1, def_type());
X        eat(')');
X
X        out_br(l2);
X        oldp = curswit;
X        curswit = &locswit;
X        locswit.caselist = NULL;
X        locswit.deflbl = -1;
X
X        stmt_b(l1);
X
X        out_br(l1);
X        def_lbl(l2);
X        gen_switch(locswit.caselist, locswit.deflbl);
X        curswit = oldp;
X        def_lbl(l1);
X        return 1;
X    case K_ELSE:
X        error("unexpected 'else'");
X        fadvnode();
X        return 0;
X    }
X}
X
XNODEP
Xparen_expr()
X{
X    NODEP np;
X    NODEP need_expr();
X
X    eat('(');
X    np = need_expr();
X    return np;
```

```
X}
X
Xbra_stmt()
X{
X    register tok;
X    NODEP np, tp;
X    NODEP opt_expr(), llook();
X
X    tok = cur->e_token;
X    fadvnode();
X    switch (tok) {
X    case K_BREAK:
X         if (debugs)
X              printf("BRK");
X         eat(';');
X         out_br(curbrk);
X         return 1;
X    case K_CONTINUE:
X         if (debugs)
X              printf("CONT ");
X         eat(';');
X         out_br(curcont);
X         return 1;
X    case K_RETURN:
X         if (debugs)
X              printf("RETURN ");
X         np = opt_expr();
X         if (np) {
X              if (funstrl)
X                   ret_stru(np);
X              else
X                   to_d0(np, funtyp);
X         }
X         out_br(funretl);
X         eat(';');
X         return 1;
X    case K_GOTO:
X         if (debugs)
X              printf("GOTO ");
X         np = cur;  advnode();
X         if (np->e_token != ID)
X              error("bad goto");
X         else {
X              tp = llook(labels, np);
X              if (tp) {
X                   freenode(np);
X              } else {
X                   tp = np;
X                   putlist(&labels, tp);
X                   tp->c_casel = new_lbl();
X              }
X              out_br(tp->c_casel);
X         }
```

```
X        eat(';');
X        return 1;
X    }
X}
X
Xlbl_stmt()
X{
X    register tok;
X    NODEP need_expr(), np;
X    int l1, i;
X
X    l1 = new_lbl();
X    tok = cur->e_token;
Xagain:
X    fadvnode();
X    switch (tok) {
X    case K_CASE:
X        if (debugs)
X            printf("CASE ");
X        np = need_expr();
X        i = conxval(np);
X        if (curswit)
X            add_case(i,l1);
X        else
X            error("'case' outside switch");
X        eat(':');
X        break;
X    case K_DEFAULT:
X        if (debugs)
X            printf("DEFAULT ");
X        if (curswit) {
X            if (curswit->deflbl >= 0)
X                error("multiple 'default'");
X            curswit->deflbl = l1;
X        } else
X            error("'default' outside switch");
X        eat(':');
X    }
X    tok = cur->e_token;    /* lookahead for more cases */
X    if (tok == K_CASE || tok == K_DEFAULT)
X        goto again;
X    def_lbl(l1);
X    return 0;
X}
X
Xasm_stmt()
X{
X    NODEP np, getexpr();
X
X    fadvnode();
X    np = getexpr();
X    if (np == NULL || np->e_token != SCON) {
X        error("bad asm() func");
```

```
X    } else {
X         out_asm(np);
X         freenode(np);
X    }
X    eat(';');
X}
X
XNODEP
Xopt_expr()
X{
X    NODE *np, *getexpr();
X
X    np = getexpr();
X    if (np) {
X         if (debugs) {
X              printf("OXPR ");
X              if (debugs > 1)
X                   printnode(np);
X         }
X    }
X    return np;
X}
X
XNODEP
Xneed_expr()
X{
X    NODE *np, *getexpr();
X
X    np = getexpr();
X    if (np) {
X         if (debugs) {
X              printf("NXPR ");
X              if (debugs > 1)
X                   printnode(np);
X         }
X    } else
X         error("need expr");
X    return np;
X}
X
Xopt_else(l1)
X{
X    int l2;
X
X    if (cur->e_token == K_ELSE) {
X         if (debugs)
X              printf("ELSE ");
X         fadvnode();
X         l2 = new_lbl();
X         out_br(l2);
X         def_lbl(l1);
X         stmt();
X         def_lbl(l2);
```

```
X    } else
X         def_lbl(l1);
X}
X
Xadd_case(val, lbl)
X{
X    NODEP np, last, p;
X
X    np = allocnode();
X    np->c_casev = val;
X    np->c_casel = lbl;
X    sprintf(np->n_name, "%d:%d", val, lbl);
X
X    last = NULL;
X    for (p = curswit->caselist; p; last=p, p=p->n_next)
X         if (p->c_casev == val) {
X              error("duplicate case");
X              return;
X         } else if (p->c_casev > val)
X              break;
X    if (last) {
X         last->n_next = np;
X         np->n_next = p;
X    } else {
X         curswit->caselist = np;
X         np->n_next = p;
X    }
X    if (debugs) {
X         printf("CASELIST\n");
X         printnode(curswit->caselist);
X    }
X}
X
Xto_d0(np, typ)
XNODEP np, typ;
X{
X    NODEP tp;
X
X    tp = allocnode();
X    tp->e_token = TCONV;
X    tp->n_tptr = typ;
X    tp->n_flags |= N_COPYT;
X    tp->n_left = np;
X    tp->e_type = E_UNARY;
X    strcpy(tp->n_name, "r cast");
X
X    do_expr(tp, IND0);
X}
X
Xret_stru(np)
XNODEP np;
X{
X    p2_expr(&np);
```

```
X    if (same_type(np->n_tptr, funtyp) == 0) {
X         error("bad struct return type");
X         return;
X    }
X    genx(np, RETSTRU);
X}
END_OF_FILE
if test 11588 -ne `wc -c <'hcc/FUN.C'`; then
    echo shar: \"'hcc/FUN.C'\" unpacked with wrong size!
fi
# end of 'hcc/FUN.C'
fi
if test -f 'hcc/G2.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/G2.C'\"
else
echo shar: Extracting \"'hcc/G2.C'\" \(11766 characters\)
sed "s/^X//" >'hcc/G2.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  g2.c
X *
X *  Generate code for binary nodes.
X */
X
X#include <stdio.h>
X#include "param.h"
X#include "nodes.h"
X#include "flags.h"
X#include "bstok.h"
X#include "gen.h"
X#include "ops.h"
X
X#define FAIL 0
X#define OKAY 1
X
Xextern int cookflags[];
X
X#define isimmed(np)    ((np)->g_flags & IMMEDID)
X#define isareg(np)     ((np)->g_token == REGVAR && (np)->g_rno >= AREG)
X
Xstruct bop {
X    char *s_op, *u_op;
X    int opflags;
X} bops[] = {
X    {"muls", "mulu",  EOPD |ASSOC},
```

```
X    {"divs", "divu",  EOPD},
X    {"divs", "divu", EOPD},
X    {"and",       "and",   EOPD|DOPE|IOPE |ASSOC},
X    {"or",        "or",    EOPD|DOPE|IOPE |ASSOC},
X    {"eor",       "eor",   DOPE|IOPE |ASSOC},
X    {"add",       "add",   EOPD|DOPE|EOPA|IOPE|AOPA|AOPD |ASSOC},
X    {"sub",       "sub",   EOPD|DOPE|EOPA|IOPE|AOPA|AOPD},
X    {"asl",       "lsl",   DOPD|QOPD|ONEOPM},
X    {"asr",       "lsr",   DOPD|QOPD|ONEOPM},
X};
X
Xchar *tstnm[] = {
X    "lt",         /* < */
X    "gt",         /* > */
X    "le",         /* <= */
X    "ge",         /* >= */
X    "eq",         /* == */
X    "ne",         /* != */
X};
X
Xint tstx[] = {
X    B_LT, B_GT, B_LE, B_GE, B_EQ, B_NE
X};
X
Xchar *utstnm[] = {
X    "cs",         /* < */
X    "hi",         /* > */
X    "ls",         /* <= */
X    "cc",         /* >= */
X    "eq",         /* == */
X    "ne",         /* != */
X};
X
Xint utstx[] = {
X    B_ULT, B_UGT, B_ULE, B_UGE, B_EQ, B_NE
X};
X
Xb_eval(np, cookie)
Xregister NODEP np;
X{
X    NODEP lp = np->n_left, rp = np->n_right;
X    NODEP tp;
X    int lcook = FORADR, rcook = FORADR;
X
X    switch (np->g_token) {      /* special cookies */
X    case DOUBLE '&':
X    case DOUBLE '|':
X         lcook = rcook = FORCC;
X         break;
X    case '?':
X         lcook = FORCC;
X         break;
X    case '(':
```

```
X         rcook = FORPUSH;
X         break;
X    case ',':
X         if (np->g_offs)    /* function args */
X              lcook = rcook = FORPUSH;
X         else {
X              lcook = FORSIDE;
X              rcook = cookie;
X         }
X         break;
X    case '=':
X         rcook = FORIMA;
X         break;
X    case '+':
X    case '-':
X         tp = rp;
X         while (tp->g_token == TCONV && tp->g_ty != ET_F)
X              tp = tp->n_left;
X         if (tp->g_token == ICON)
X              lcook = FORIMA;
X         break;
X    }
X
X    if (np->g_type == EV_LR) {
X         if (eval(lp,lcook) == FAIL)
X              return FAIL;
X         freetmps(lp);
X         if (eval(rp,rcook) == FAIL)
X              return FAIL;
X         freetmps(rp);
X    } else if (np->g_type == EV_RL) {
X         if (eval(rp,rcook) == FAIL)
X              return FAIL;
X         freetmps(rp);
X         if (eval(lp,lcook) == FAIL)
X              return FAIL;
X         freetmps(lp);
X    } else {  /* EV_LRSEP */
X         if (eval(lp,lcook) == FAIL)
X              return FAIL;
X         freetmps(lp);
X         free1(NULL, lp);
X         if (eval(rp,rcook) == FAIL)
X              return FAIL;
X         freetmps(rp);
X    }
X    return b_sube(np, cookflags[cookie]);
X}
X
Xb_sube(np, flags)
Xregister NODEP np;
X{
X    NODEP lp = np->n_left, rp = np->n_right;
```

```
X    register int i, r;
X    int argsize;
X    char buf[40];
X
X    if (isassign(np->g_token))
X         return as_eval(np);
X
X    switch (np->g_token) {
X    case '=':
X         if (specasn(np, flags) || strasn(np))
X              return OKAY;
X         inherit(np);
X         addcode(np, "\tmove.S\t>A,<A\n");
X         return OKAY;
X
X    case '(':
X         argsize = argmod(rp);
X         free1(NULL,rp);
X         if (np->g_ty == ET_A) {     /* struct returned */
X              frc_ral(AREG);
X              indir(np, AREG);
X         } else {
X              frc_ral(0);
X              retreg(np, 0);
X         }
X         sprintf(buf, "\tjsr\t<A\n\tadd.w\t#%d,sp\n", argsize);
X         addcode(np, buf);
X         return OKAY;
X
X    case ',':
X         if (np->g_offs == 0)    /* normal ',' */
X              rinherit(np);
X         return OKAY;
X
X    case DOUBLE '&':
X         free1(NULL, rp);
X         r = ralloc(0);
X         retreg(np, r);
X         holdlbls(np);
X         np->g_betw = iscc(lp) ? "<FL1\n" :
X              "<Q\tbeq\tL1\n";
X         addcode(np, iscc(rp) ? ">FL1\n" :
X              ">Q\tbeq\tL1\n");
X         addcode(np, "\tmoveq\t#1,A\n");
X         addcode(np, "\tbra\tL2\nL1:\tclr\tA\nL2:\n");
X         return OKAY;
X
X    case DOUBLE '|':
X         free1(NULL, rp);
X         r = ralloc(0);
X         retreg(np, r);
X         holdlbls(np);
X         np->g_betw = iscc(lp) ? "<TL1\n" :
```

```
X             "<Q\tbne\tL1\n";
X         addcode(np, iscc(rp) ? ">TL1\n" :
X             ">Q\tbne\tL1\n");
X         addcode(np, "\tclr\tA\n");
X         addcode(np, "\tbra\tL2\nL1:\tmoveq\t#1,A\nL2:\n");
X         return OKAY;
X
X    case '?':
X         rinherit(np);
X         rinhlbls(np);
X         np->g_betw = iscc(lp) ? "<FL1\n" : "<Q\tbeq\tL1\n";
X         return OKAY;
X
X    case ':':
X         free1(NULL, rp);
X         r = ralloc(0);
X         retreg(np, r);
X         holdlbls(np);
X         np->g_betw = same_a(np, lp) ?
X             "\tbra\tL2\nL1:\n"  :
X             "\tmove.S\t<A,A\n\tbra\tL2\nL1:\n";
X         if (!same_a(np, rp))
X             addcode(np, "\tmove.S\t>A,A\n");
X         addcode(np, "L2:\n");
X         return OKAY;
X
X    case '<':
X         i = 0;  goto dotst;
X    case '>':
X         i = 1;  goto dotst;
X    case LTEQ:
X         i = 2;  goto dotst;
X    case GTEQ:
X         i = 3;  goto dotst;
X    case DOUBLE '=':
X         i = 4;  goto dotst;
X    case NOTEQ:
X         i = 5;
Xdotst:
X         fix_cmp(np, EOPD|EOPA|IOPE|AOPA|AOPD);
X         if (flags & CC_OK) {
X             np->g_token = (lp->g_ty == ET_U ?
X             utstx[i] : tstx[i]) + BR_TOK;
X         } else {
X             strcpy(np->n_name, lp->g_ty == ET_U ?
X             utstnm[i] : tstnm[i]);
X             r = ralloc(0);
X             retreg(np, r);
X             addcode(np, "\tsN\tA\n\tand.w\t#1,A\n");
X         }
X         return OKAY;
X
X    case '*':
```

```
X        return fixmul(np, bops[0].opflags);
X    case '/':
X        return fixdiv(np, bops[1].opflags);
X    case '%':
X        return fixmod(np, bops[2].opflags);
X    case '&': i = 3;  goto doop;
X    case '|': i = 4;  goto doop;
X    case '^': i = 5;   goto doop;
X    case '+':
X        if (optadd(np, flags, 1))
X            return OKAY;
X            i = 6;  goto doop;
X    case '-':
X        if (optadd(np, flags, -1))
X            return OKAY;
X            i = 7;  goto doop;
X    case DOUBLE '<':i = 8;  goto doop;
X    case DOUBLE '>':i = 9;
Xdoop:
X        strcpy(np->n_name, np->g_ty == ET_U ?
X            bops[i].u_op : bops[i].s_op);
X        r = fix2ops(np, bops[i].opflags);
X        cc_hack(np);
X        return r;
X    case FIELDAS:
X        return fldasn(np, flags);
X    default:
X        printf("Weird b_eval %s ", np->n_name);
X        return FAIL;
X    }
X}
X
Xas_eval(np)
Xregister NODEP np;
X{
X    NODEP rp = np->n_right;
X    register int op, i, r;
X
X    rp = np->n_right;
X
X    op = np->g_token;
X    op -= ASSIGN 0;
X    switch (op) {
X
X    /* these get unfolded now */
X    case '*':
X            return fixamul(np, bops[0].opflags);
X    case '/':
X            return fixadiv(np, bops[1].opflags);
X    case '%':
X            return fixamod(np, bops[2].opflags);
X    case '&': i = 3;  goto doop;
X    case '|': i = 4;  goto doop;
```

```
X    case '^': i = 5;   goto doop;
X    case '+': i = 6;  goto doop;
X    case '-': i = 7;  goto doop;
X    case DOUBLE '<':i = 8;  goto doop;
X    case DOUBLE '>':i = 9;
Xdoop:
X        strcpy(np->n_name, np->g_ty == ET_U ?
X            bops[i].u_op : bops[i].s_op);
X        r = fix_asn(np, bops[i].opflags);
X        cc_hack(np);
X        return r;
X
X    default:
X        printf("Weird as_eval %s ", np->n_name);
X        return FAIL;
X    }
X}
X
Xrinherit(np)
Xregister NODEP np;
X{
X    register NODEP rp = np->n_right;
X
X    np->g_token = rp->g_token;
X    np->g_offs = rp->g_offs;
X    np->g_rno = rp->g_rno;
X    np->g_flags |= RCHILDNM | (rp->g_flags & IMMEDID);
X}
X
Xargmod(np)
Xregister NODEP np;
X{
X    int size = 0;
X
X    if (np->g_token == ',') {
X        np->g_type = EV_RL;
X        size += argmod(np->n_right);
X        size += argmod(np->n_left);
X        return size;
X    }
X    size += onearg(np);
X    return size;
X}
X
Xonearg(np)
Xregister NODEP np;
X{
X    int rv;
X
X    /* hack small ICON */
X    if (np->g_sz == 1 && np->g_token == ICON)
X        np->g_sz = 2;
X    /* hack push of 0 */
```

```
X    if (np->g_token == ICON && np->g_offs == 0 && isimmed(np)) {
X         addcode(np, "\tclr.S\t-(sp)\n");
X         return (int)np->g_sz;
X    }
X    /* hack push of #OREG */
X    if (np->g_token == OREG && isimmed(np)) {
X         np->g_flags &= ~IMMEDID;
X         addcode(np, "\tpea\tA\n");
X         return 4;
X    }
X
X    if (np->g_ty == ET_A) {
X         rv = strpush(np);
X         freetmps(np);
X         free1(NULL,np);
X         return rv;
X    }
X
X    switch (np->g_sz) {
X    case 1:
X         addcode(np, (np->g_ty == ET_U) ?
X            "\tclr.w\td0\n\tmove.b\tA,d0\n\tmove.w\td0,-(sp)\n" :
X            "\tmove.b\tA,d0\n\text.w\td0\n\tmove.w\td0,-(sp)\n" );
X         return 2;
X    case 2:
X         addcode(np,
X            "\tmove.w\tA,-(sp)\n");
X         return 2;
X    default:
X         addcode(np,
X            "\tmove.l\tA,-(sp)\n");
X         return 4;
X    }
X}
X
X#define MAXD DRV_START
X#define MAXA (ARV_START-AREG)
X#define NEEDALL   (MAXA*AREG + MAXD)
X
Xorder(np)
Xregister NODEP np;
X{
X    int l, r;
X
X    switch (np->g_type) {
X    case E_BIN:
X         order(np->n_right);
X         r = np->n_right->g_needs;
X    case E_UNARY:
X         order(np->n_left);
X         l = np->n_left->g_needs;
X         break;
X    default: /* leaf */
```

```
X         np->g_type = EV_NONE;
X         np->g_needs = 0;
X         return;
X    }
X
X    if (np->g_type == E_UNARY) {
X         switch (np->g_token) {
X         case STAR:
X         case UNARY '&':
X              np->g_needs = merge(l,AREG);
X              break;
X         case '(':
X              np->g_needs = NEEDALL;
X              break;
X         case POSTINC:
X         case POSTDEC:
X         case '!':
X         case TCONV:
X              np->g_needs = merge(l,1);
X              break;
X         case '.':
X              if (np->g_fldw) {
X                   np->g_needs = merge(l,1);
X                   break;
X              }
X              /* else fall through */
X         default:
X              np->g_needs = l;
X         }
X         np->g_type = EV_LEFT;
X         return;
X    }
X
X/* at this point, have binary node */
X
X    switch (np->g_token) {
X    case DOUBLE '&':
X    case DOUBLE '|':
X    case '?':
X    case ':':
X         /* always left-right, no extra regs */
X         np->g_type = EV_LRSEP;
X         np->g_needs = merge(1, merge(l,r));
X         return;
X    case ',':
X         np->g_needs = merge(l, r);
X         np->g_type = EV_LRSEP;
X         return;
X    case '(':
X         np->g_needs = NEEDALL;
X         break;
X    case '^':
X    case DOUBLE '<':
```

```
X    case DOUBLE '>':
X    case ASSIGN '/':
X    case ASSIGN DOUBLE '<':
X    case ASSIGN DOUBLE '>':
X         np->g_needs = merge(bin(l,r), 2);
X         break;
X    default:
X         if (isassign(np->g_token) || np->g_token == '=')
X              np->g_needs = merge(rbin(l,r), 1);
X         else
X              np->g_needs = merge(bin(l,r), 1);
X    }
X
X    if (isassign(np->g_token) || np->g_token == '=')
X         np->g_type = EV_RL;     /* NO PUSHER's on L */
X    else
X         np->g_type = worst_1st(l, r);
X    flag_saves(np, l, r);
X}
X
Xflag_saves(np, l, r)
XNODEP np;
X{
X    NODEP *cpp;
X    register int other;
X
X    if (np->g_type == EV_LR) {
X         cpp = &np->n_left;
X         other = r;
X    } else {
X         cpp = &np->n_right;
X         other = l;
X    }
X    if ((other & 7) >= MAXD || (other/AREG) >= MAXA)
X         addtmp(np, cpp);
X}
X
Xaddtmp(np, cpp)
XNODEP np, *cpp;
X{
X    NODEP cp, tp;
X    NODEP copyone();
X
X    cp = *cpp;
X    tp = copyone(cp);
X    tp->n_left = cp;
X    *cpp = tp;
X    tp->g_token = PUSHER;
X    strcpy(tp->n_name, "pusher");
X    tp->g_type = EV_LEFT;
X}
X
Xworst_1st(l,r)
```

```
X{
X    int ld, rd;
X
X    ld = l & 7;
X    rd = r & 7;
X    if (rd > ld)
X         return EV_RL;
X    if (r > l)
X         return EV_RL;
X    return EV_LR;
X}
X
Xbin(l,r)
X{
X    int la, ra, na;
X    int ld, rd, nd;
X    int rfirst;
X
X    la = l/AREG;
X    ra = r/AREG;
X    ld = l & 7;
X    rd = r & 7;
X    rfirst = (ra > la) || (rd > ld);
X    return rfirst ? rbin(l,r) : rbin(r,l);
X}
X
Xrbin(last,first)
X{
X    int la, fa, na;
X    int ld, fd, nd;
X
X    la = last/AREG;
X    fa = first/AREG;
X    ld = last & 7;
X    fd = first & 7;
X    na = fa > la ? fa : la + (fa ? 1 : 0);
X    if (na > MAXA)
X         na = MAXA;
X    nd = fd > ld ? fd : ld + (fd ? 1 : 0);
X    if (nd > MAXD)
X         nd = MAXD;
X    return na*AREG + nd;
X}
X
Xmerge(need, have)
X{
X    int na, nd, ha, hd, xa, xd;
X
X    na = need/AREG;
X    ha = have/AREG;
X    nd = need & 7;
X    hd = have & 7;
X    xa = na > ha ? na : ha;
```

```
X    xd = nd > hd ? nd : hd;
X    return xa*AREG + xd;
X}
X
Xholdlbls(np)
XNODEP np;
X{
X    np->g_bsize = new_lbl();
X    new_lbl();
X}
X
Xrinhlbls(np)
XNODEP np;
X{
X    np->g_bsize = np->n_right->g_bsize;
X}
X
X/* limited version of same address check
X    assume one of these is a temp register */
Xsame_a(p1, p2)
XNODEP p1, p2;
X{
X    if (p1->g_token != p2->g_token)
X         return 0;
X    if (p1->g_rno != p2->g_rno)
X         return 0;
X    return 1;
X}
X
Xoptadd(np, flags, sign)
Xregister NODEP np;
X{
X    NODEP lp = np->n_left, rp = np->n_right;
X
X    if (rp->g_token != ICON)
X         return 0;
X    if (isimmed(lp) && isimmed(rp)) {
X         switch (lp->g_token) {
X         case OREG:
X         case ONAME:
X              inherit(np);
X              if (sign == -1)
X                   rp->g_offs = -rp->g_offs;
X              np->g_offs += rp->g_offs;
X
X              if ((flags & IMMA_OK) == 0)
X                   imm_oreg(np);
X              return 1;
X         default:
X              return 0;
X         }
X    }
X    return 0;
```

```
X}
X
Xiscc(np)
XNODEP np;
X{
X    return (np->g_token >= BR_TOK) || (np->g_flags & SIDE_CC);
X}
X
Xcc_hack(np)
XNODEP np;
X{
X    if (isareg(np))
X        return;
X    np->g_flags |= SIDE_CC;
X}
X
Xcctok(np)
XNODEP np;
X{
X    if (np->g_token >= BR_TOK)
X        return np->g_token - BR_TOK;
X    if (np->g_flags & SIDE_CC)
X        return B_NE;
X    printf("cctok error ");
X    return 0;
X}
END_OF_FILE
if test 11766 -ne `wc -c <'hcc/G2.C'`; then
    echo shar: \"'hcc/G2.C'\" unpacked with wrong size!
fi
# end of 'hcc/G2.C'
fi
if test -f 'hcc/GEN.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/GEN.C'\"
else
echo shar: Extracting \"'hcc/GEN.C'\" \(11959 characters\)
sed "s/^X//" >'hcc/GEN.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  gen.c
X *
X *  Generate code.
X *  Includes main routine and code generation for unary nodes
X *  and leafs.
X */
```

```
X
X#include <stdio.h>
X#include "param.h"
X#include "bstok.h"
X#include "tytok.h"
X#include "flags.h"
X#include "nodes.h"
X#include "gen.h"
X
XNODEP strsave;
Xint cctest;
Xstatic int reserve;
Xstatic int tmpused;
X
Xextern xflags[];
X#define debug xflags['g'-'a']
X
X#define FAIL 0
X#define OKAY 1
X
X#define isimmed(np)    ((np)->g_flags & IMMEDID)
X#define isareg(np)     ((np)->g_token == REGVAR && (np)->g_rno >= AREG)
X#define isdreg(np)     ((np)->g_token == REGVAR && (np)->g_rno < AREG)
X#define istdreg(np)    ((np)->g_token == REGVAR && (np)->g_rno < DRV_START)
X
Xint cookflags[] = {
X    0,
X    NOVAL_OK|CC_OK|IMMA_OK,     /* FORSIDE */
X    IMMA_OK,           /* FORPUSH */
X    CC_OK,                  /* FORCC */
X    IMMA_OK,           /* FORIMA */
X    0,                 /* FORADR */
X    IMMA_OK,           /* FORINIT */
X    0,                 /* IND0 */
X    0,                 /* RETSTRU */
X};
X
Xgenx(np, cookie)
Xregister NODEP np;
X{
X    extern nmerrors;
X    int rv;
X    extern NODE *blktab;
X
X    if (np == NULL) return;
X    if (nmerrors)
X         goto bad;
X    if (debug) {
X         printf("GEN enter");
X         printnode(np);
X    }
X
X    untype(np);
```

```
X    if (debug>1) {
X         printf("after UNTYPE");
X         printnode(np);
X    }
X
X    tmpused = 0;
X    gunk(np);
X    if (tmpused && tmpused > blktab->b_tsize)
X         blktab->b_tsize = tmpused;
X    if (debug > 1) {
X         printf("after gunk");
X         printnode(np);
X    }
X    order(np);
X
X    reserve = 0;
X    rv = eval(np, cookie);
X    if (rv == OKAY && debug) {
X         printf("GEN exit");
X         codeprint(np);
X    }
X    if (rv == OKAY)
X         rv = cookmon(np, cookie);
X    else
X         error("couldnt eval node");
X    freenode(np);
X    return rv;
Xbad:
X    freenode(np);
X    return FAIL;
X}
X
Xeval(np, cookie)
Xregister NODEP np;
X{
X    int rv;
X
X    np->g_r1 = np->g_r2 = -1;
X
X    if (np == NULL) return FAIL;
X
X    switch (np->g_type) {
X
X    default:
X         rv = b_eval(np, cookie);
X         /* already did freetmps */
X         free1(np, np->n_left);
X         free1(np, np->n_right);
X         break;
X
X    case EV_LEFT:
X         rv = u_eval(np, cookie);
X         freetmps(np);
```

```
X         free1(np, np->n_left);
X         break;
X
X    case EV_NONE:
X         rv = l_eval(np);
X         break;
X    }
X    return rv;
X}
X
Xu_eval(np, cookie)
Xregister NODEP np;
X{
X    int subcook = FORADR;
X
X    switch (np->g_token) {
X    case STAR:
X         subcook = FORIMA;
X         break;
X    case '!':
X         subcook = FORCC;
X         break;
X    }
X
X    if (eval(np->n_left, subcook) == FAIL)
X         return FAIL;
X    return u_sube(np, cookflags[cookie]);
X}
X
Xu_sube(np, flags)
Xregister NODEP np;
X{
X    register NODEP lp = np->n_left;
X    long offs;
X    int i;
X
X    switch (np->g_token) {
X    case '.':
X         if (np->g_fldw)
X              return fldget(np, flags);
X         offs = np->g_offs;
X         inherit(np);
X         np->g_offs += offs;
X         return OKAY;
X    case STAR:
X         if (isimmed(lp)) {
X              inherit(np);
X              np->g_flags &= ~IMMEDID;
X         } else if (isareg(lp)) {
X              indir(np, lp->g_rno);
X         } else {  /* NEED A temp */
X              if (lp->g_token == OREG && istemp(lp->g_rno))
X                   i = lp->g_rno;
```

```
X             else
X                  i = ralloc(AREG);
X             addcode(np, "\tmove.l\t<A,R0\n");
X             indir(np, i);
X        }
X        return OKAY;
X    case UNARY '&':
X        if (isimmed(lp))
X             warn("& ignored");
X        else if (lp->g_token == REGVAR)
X             return FAIL;
X        inherit(np);
X        np->g_flags |= IMMEDID;
X        if ((flags & IMMA_OK) == 0)
X             imm_oreg(np);
X        return OKAY;
X    case '~':
X        utemp(np);
X        addcode(np, "\tnot.S\tA\n");
X        cc_hack(np);
X        return OKAY;
X    case UNARY '-':
X        utemp(np);
X        addcode(np, "\tneg.S\tA\n");
X        cc_hack(np);
X        return OKAY;
X    case TCONV:
X        castgen(np);
X        return OKAY;
X    case PUSHER:  /* must not be on left of assign or asn-op */
X        if ((lp->g_token != OREG && lp->g_token != REGVAR) ||
X             istemp(lp->g_rno) == 0) {
X             inherit(np);
X             return OKAY;
X        }
X        if (lp->g_ty == ET_A)
X             strxpush(np);
X        else
X             addcode(np, "\tmove.S\t<A,-(sp)\n");
X        return OKAY;
X    case '(':
X        if (np->g_ty == ET_A) {     /* struct returned */
X             frc_ral(AREG);
X             indir(np, AREG);
X        } else {
X             frc_ral(0);
X             retreg(np, 0);
X        }
X        addcode(np, "\tjsr\t<A\n");
X        return OKAY;
X    case DOUBLE '+':
X        holdcon(np);
X        inherit(np);
```

```
X         addcode(np, "\tadd.S\t#K,A\n");
X         cc_hack(np);
X         return OKAY;
X    case DOUBLE '-':
X         holdcon(np);
X         inherit(np);
X         addcode(np, "\tsub.S\t#K,A\n");
X         cc_hack(np);
X         return OKAY;
X    case POSTINC:
X         if ((flags & NOVAL_OK) == 0) {
X              i = ralloc(0);
X              retreg(np, i);
X              addcode(np, "\tmove.S\t<A,A\n");
X         }
X         addcode(np, "\tadd.S\t#O,<A\n");
X         return OKAY;
X    case POSTDEC:
X         if ((flags & NOVAL_OK) == 0) {
X              i = ralloc(0);
X              retreg(np, i);
X              addcode(np, "\tmove.S\t<A,A\n");
X         }
X         addcode(np, "\tsub.S\t#O,<A\n");
X         return OKAY;
X    case CMPBR:
X         i = ralloc(0);
X         retreg(np, i);
X         addcode(np, "\tsN\tA\n\tand.w\t#1,A\n");
X         cc_hack(np);
X         return OKAY;
X    case '!':
X         if (flags & CC_OK) {
X              if (iscc(lp)) {
X                   i = cctok(lp);
X                   i = (i&1) ? i+1 : i-1; /* reverse truth */
X              } else {
X                   i = B_EQ;
X                   addcode(np, "<Q");
X              }
X              np->g_token = i + BR_TOK;
X         } else {
X              if (istdreg(lp))
X                   i = lp->g_rno;
X              else
X                   i = ralloc(0);
X              retreg(np, i);
X              if (iscc(lp)) {
X                   addcode(np, "<EA\n\tand.w\t#1,A\n");
X              } else {
X                   addcode(np, "<Q");
X                   addcode(np, "\tseq\tA\n\tand.w\t#1,A\n");
X              }
```

```
X         }
X         return OKAY;
X    default:
X         printf("Weird u_eval %s ", np->n_name);
X         return FAIL;
X    }
X}
X
Xholdcon(np)
XNODEP np;
X{
X    np->g_bsize = np->g_offs;
X}
X
Xretreg(np, rno)
XNODEP np;
X{
X    np->g_token = REGVAR;
X    np->g_rno = rno;
X}
X
Xindir(np, rno)
Xregister NODEP np;
X{
X    np->g_token = OREG;
X    np->g_offs = 0;
X    np->g_rno = rno;
X}
X
Xinherit(np)
Xregister NODEP np;
X{
X    NODEP lp = np->n_left;
X
X    np->g_token = lp->g_token;
X    np->g_offs = lp->g_offs;
X    np->g_rno = lp->g_rno;
X    np->g_flags |= CHILDNM | (lp->g_flags & IMMEDID);
X}
X
Xcookmon(np, cookie)
Xregister NODEP np;
X{
X    extern FILE *output;
X
X    if (np == NULL) return FAIL;
X
X    switch (cookie) {
X    case FORCC:
X         if (iscc(np)) {
X              outcode(np);
X              cctest = cctok(np);
X         } else {
```

```
X             if (np->g_token == ICON && isimmed(np)) {
X                  cctest = np->g_offs ? B_YES : B_NO;
X                  return OKAY;
X             }
X             outcode(np);
X             outsub("Q", np);
X             cctest = B_NE;
X        }
X        return OKAY;
X    case FORINIT:
X        if (anycode(np) == 0 && (np->g_flags & IMMEDID)) {
X             out_a(np, output);
X             return OKAY;
X        }
X        error("bad INIT expr");
X        return FAIL;
X    case IND0:
X        outcode(np);
X        if (np->g_token != REGVAR ||
X             np->g_rno != 0)
X             if (np->g_token == ICON && np->g_offs == 0 &&
X                  isimmed(np))
X                  outsub("\tclr.S\td0\n", np);
X             else
X                  outsub("\tmove.S\tA,d0\n", np);
X        return OKAY;
X    case RETSTRU:
X        freetmps(np);
X        strret(np);
X        outcode(np);
X        return OKAY;
X    }
X    outcode(np);
X    return OKAY;
X}
X
Xanycode(np)
Xregister NODEP np;
X{
X    if (np->g_code)
X        return 1;
X    switch (np->g_type) {
X    case EV_NONE:
X        return 0;
X    case EV_LEFT:
X        return anycode(np->n_left);
X    case EV_RIGHT:
X        return anycode(np->n_right);
X    case EV_LR:
X    case EV_RL:
X        return anycode(np->n_left) || anycode(np->n_right);
X    }
X}
```

```
X
Xl_eval(np)
Xregister NODEP np;
X{
X    int l1;
X
X    switch (np->g_token) {
X    case ID:
X         switch (np->g_sc) {
X         default: /* EXTERN or HERE */
X              np->g_token = ONAME;
X              np->g_offs = 0;
X              if (np->n_name[0] != '%')
X                   nnmins(np, "_");
X              else /* hack for inline name */
X                   strcpy(np->n_name, &np->n_name[1]);
X              return OKAY;       /* dont free n_nmx */
X         case K_STATIC:
X              sprintf(np->n_name, "L%d", (int)np->g_offs);
X              np->g_offs = 0;
X              np->g_token = ONAME;
X              break;
X         case K_AUTO:
X              np->g_rno = FRAMEP;
X              np->g_token = OREG;
X              break;
X         case K_REGISTER:
X              np->g_token = REGVAR;
X              break;
X         }
X         if (np->n_nmx) {
X              freenode(np->n_nmx);
X              np->n_nmx = NULL;
X         }
X         return OKAY;
X    case ICON:
X         np->g_flags |= IMMEDID;
X         return OKAY;
X    case FCON:
X         np->g_flags |= IMMEDID;
X         return OKAY;
X    case SCON:
X         np->g_flags |= IMMEDID;
X         np->g_token = ONAME;
X         l1 = new_lbl();
X         save_scon(np, l1);
X         sprintf(np->n_name, "L%d", l1);
X         return OKAY;
X    case OREG:
X         return OKAY;
X    }
X    return FAIL;
X}
```

```
X
Xsave_scon(np, lbl)
XNODEP np;
X{
X    NODEP tp, copyone();
X
X    tp = copyone(np);
X    tp->g_offs = lbl;
X    if (np->n_nmx) {
X         freenode(np->n_nmx);
X         np->n_nmx = NULL;
X    }
X    putlist(&strsave, tp);
X}
X
Xutemp(np)
XNODEP np;
X{
X    NODEP lp = np->n_left;
X    int i;
X
X    if (lp->g_token == REGVAR &&
X        istemp(lp->g_rno)) {
X         inherit(np);
X         return;
X    }
X    i = ralloc(0);
X    retreg(np, i);
X    addcode(np, "\tmove.S\t<A,A\n");
X}
X
Xfreetmps(np)
Xregister NODEP np;
X{
X    if (np->g_r1 != -1)
X         rfree(np->g_r1);
X    if (np->g_r2 != -1)
X         rfree(np->g_r2);
X}
X
Xfree1(np, cp)
XNODEP np, cp;
X{
X    int cr;
X
X    if (cp->g_token != OREG && cp->g_token != REGVAR)
X         return;
X    cr = cp->g_rno;
X    if (np && cr == np->g_rno &&
X         (np->g_token == OREG || np->g_token == REGVAR))
X         return;
X    if (istemp(cr))
X         rfree(cr);
```

```
X}
X
Xistemp(rno)
X{
X    return (rno < DRV_START ||
X         (rno >= AREG && rno < ARV_START));
X}
X
Xrfree(rno)
X{
X    reserve &= ~(1<<rno);
X}
X
Xfrc_ral(rno)
X{
X    int i;
X
X    i = (1<<rno);
X    if (reserve & i)
X         error("rno reserved! ");
X    reserve |= i;
X}
X
Xtempr(np, type)
XNODEP np;
X{
X    int i;
X
X    i = ralloc(type);
X    if (np->g_r1 == -1)
X         np->g_r1 = i;
X    else
X         np->g_r2 = i;
X    return i;
X}
X
Xralloc(type)
X{
X    int starti, endi;
X    register int i;
X
X    if (type == AREG) {
X         starti = AREG;
X         endi = ARV_START;
X    } else {
X         starti = 0;
X         endi = DRV_START;
X    }
X    for (i=starti; i<endi; i++)
X         if ((reserve & (1<<i)) == 0) {
X              reserve |= (1<<i);
X              return i;
X         }
```

```
X    error("Compiler failure - rallo");
X    return -1;
X}
X
Xtmp_alloc(sz)
X{
X    extern NODE *blktab;
X
X    tmpused += sz;
X    return blktab->b_size + tmpused;
X}
X
X/* fixes nodes with no code or aX is temp that are #d(aX) */
Ximm_oreg(np)
XNODEP np;
X{
X    char *regnm(), buf[30];
X    int i;
X
X    if (np->g_token != OREG)
X         return;
X    if ((np->g_flags & IMMEDID) == 0)
X         return;
X    np->g_flags &= ~IMMEDID;
X    if (np->g_offs == 0) {      /* #(a0) -> a0 */
X         np->g_token = REGVAR;
X         return;
X    }
X    if (istemp(np->g_rno)) {
X         holdcon(np);
X         addcode(np, "\tadd\t#K,A\n");
X         np->g_token = REGVAR;
X         return;
X    }
X    sprintf(buf, "\tlea\t%d(%s),A\n", (int)np->g_offs, regnm(np->g_rno));
X    addcode(np, buf);
X    i = ralloc(AREG);
X    retreg(np, i);
X}
X
Xcastgen(tp)
Xregister NODEP tp;
X{
X    register NODEP np = tp->n_left;
X    int sz = tp->g_sz;
X    int i;
X
X    if (np->g_token == ICON && isimmed(np)) {
X         if (tp->g_ty == ET_F) {
X              tp->g_token = FCON;
X              *(float *)&tp->g_offs = (float)np->g_offs;
X              tp->g_flags |= IMMEDID;
X         } else {
```

```
X             inherit(tp);
X             i_exp(tp, np->g_sz, np->g_ty);
X             squish(tp);
X        }
X    } else if (np->g_token == FCON && isimmed(np)) {
X        if (tp->g_ty != ET_F) {
X             tp->g_token = ICON;
X             tp->g_offs = (long)*(float *)&np->g_offs;
X             tp->g_flags |= IMMEDID;
X             squish(tp);
X        } else {
X             inherit(tp);
X        }
X    } else if (sz > np->g_sz) {
X        if (np->g_ty == ET_U) {
X             i = ralloc(0);
X             retreg(tp, i);
X             addcode(tp, "\tclr.S\tA\n\tmove.<S\t<A,A\n");
X        } else {
X             if (isdreg(np)) {
X                  inherit(tp);
X             } else {
X                  i = ralloc(0);
X                  retreg(tp, i);
X                  addcode(tp, "\tmove.<S\t<A,A\n");
X             }
X             if (sz == 4 && np->g_sz == 1)
X                  addcode(tp, "\text.w\tA\n\text.l\tA\n");
X             else
X                  addcode(tp, "\text.S\tA\n");
X        }
X        return;
X    }
X    else if (sz < np->g_sz) {
X        switch (np->g_token) {
X        case ONAME:
X        case OREG:
X             if (isimmed(np)) {
Xsmfudge:
X                  i = ralloc(0);
X                  retreg(tp, i);
X                  addcode(tp, "\tmove.<S\t<A,A\n");
X                  return;
X             }
X             inherit(tp);
X             tp->g_offs = np->g_offs + (np->g_sz - sz);
X             break;
X        case REGVAR:
X             if (sz == 1 && np->g_rno >= AREG)
X                  goto smfudge;
X             /* fall through */
X        default:
X             inherit(tp);
```

```
X        }
X    } else
X        inherit(tp);
X}
X
Xsquish(np)
XNODEP np;
X{
X    int neg;
X
X    neg = (np->g_ty == ET_S && np->g_offs < 0);
X
X    switch (np->g_sz) {
X    case 1:
X        if (neg)
X            np->g_offs |= 0xffffff00L;
X        else
X            np->g_offs &= 0xff;
X        break;
X    case 2:
X        if (neg)
X            np->g_offs |= 0xffff0000L;
X        else
X            np->g_offs &= 0xffffL;
X        break;
X    }
X}
X
Xi_exp(np, osz, oty)
XNODEP np;
X{
X    long l;
X
X    if (oty == ET_S && osz < np->g_sz) {
X        l = np->g_offs;
X        switch (osz) {
X        case 1:
X            l = (char) l;
X            break;
X        case 2:
X            l = (short) l;
X            break;
X        }
X        np->g_offs = l;
X    }
X}
END_OF_FILE
if test 11959 -ne `wc -c <'hcc/GEN.C'`; then
    echo shar: \"'hcc/GEN.C'\" unpacked with wrong size!
fi
# end of 'hcc/GEN.C'
fi
echo shar: End of archive 5 \(of 9\).
```

```
cp /dev/null ark5isdone
MISSING=""
for I in 1 2 3 4 5 6 7 8 9 ; do
    if test ! -f ark${I}isdone ; then
     MISSING="${MISSING} ${I}"
    fi
done
if test "${MISSING}" = "" ; then
    echo You have unpacked all 9 archives.
    rm -f ark[1-9]isdone ark[1-9][0-9]isdone
else
    echo You still need to unpack the following archives:
    echo "        " ${MISSING}
fi
##  End of shell archive.
exit 0


From -982306104390795317
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,a4dab9691fdba901,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-25 10:15:19 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!
gem.mps.ohio-state.edu!apple!sun-barr!newstop!sun!sunfedcomm!grapevine!
koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i097: sozobon1.2 -- Update to Sozobon C compiler  part06/09
Keywords: shar, C
Message-ID: <34610@grapevine.uucp>
Date: 25 Oct 89 17:15:19 GMT
Sender: daemon@sunfedcomm
Reply-To: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Lines: 1924
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Posting-number: Volume 2, Issue 97
Archive-name: sozobon1.2/part06

#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of archive 6 (of 9)."
# Contents:  hcc/OUT_ST.C top/BRANCH.C top/PEEP3.C
# Wrapped by koreth@panarthea on Tue Oct 24 18:40:46 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'hcc/OUT_ST.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/OUT_ST.C'\"
```

```
else
echo shar: Extracting \"'hcc/OUT_ST.C'\" \(12658 characters\)
sed "s/^X//" >'hcc/OUT_ST.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X * 
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  out.c
X *
X *  Code generation output routines.
X */
X
X#include <stdio.h>
X#include "param.h"
X#include "nodes.h"
X#include "flags.h"
X#include "bstok.h"
X#include "tytok.h"
X#include "gen.h"
X
X#if dLibs
X#include <ctype.h>
X#endif
X
X#if MMCC
Xoverlay "pass2"
X#endif
X
X#if CC68
XFILE *fopen();
X#endif
X
X#if NEEDBUF
Xchar my_obuf[BUFSIZ];
X#endif
X
X#define T_SEG     0
X#define D_SEG     1
X#define B_SEG     2
X
X#define TO_TEXT   to_seg(T_SEG)
X#define TO_DATA   to_seg(D_SEG)
X#define TO_BSS    to_seg(B_SEG)
X
X#define isareg(np)     ((np)->g_token == REGVAR && (np)->g_rno >= AREG)
X
Xextern FILE *output;
```

```
X
Xstatic int in_seg;
Xstatic int lblnum;
Xstatic int dat_size;
X
Xout_start(outs)
Xchar *outs;
X{
X    register int len;
X    char suff;
X#ifdef MINIX
X    extern char tmpdir[];
X#endif
X    char oname[128];
X
X    len = strlen(outs);
X    if (len >= 2 && outs[len-2] == '.') {
X         suff = outs[len-1];
X         if (suff != 'c' && suff != 'C')
X              fatals("Invalid suffix", outs);
X#ifdef MINIX
X         sprintf(oname, "%s/%s", tmpdir, outs);
X         len = strlen(oname);
X#else
X         strcpy(oname, outs);
X#endif
X         oname[len-1] = 's';
X         output = fopen(oname, "w");
X         if (output == NULL)
X              fatals("Cant open", oname);
X#if NEEDBUF
X         setbuf(output, my_obuf);
X#endif
X    } else
X         output = stdout;
X
X    in_seg = -1;
X    lblnum = 0;
X    dat_size = 0;
X}
X
Xout_end()
X{
X    if (output != stdout)
X         fclose(output);
X}
X
Xstatic char *sg_go[] = {
X    ".text",
X    ".data",
X    ".bss"
X};
X
```

```
Xto_text()
X{
X    TO_TEXT;
X}
X
Xto_seg(sg)
X{
X    if (sg == in_seg)
X         return;
X    fprintf(output, "\t%s\n", sg_go[sg]);
X    in_seg = sg;
X}
X
Xo_aln(x)
X{
X    if (x && (dat_size & 1)) {
X         dat_size++;
X         TO_DATA;
X         fprintf(output, "\t.even\n");
X    }
X}
X
Xchar *rnms[] = {
X    "d0", "d1", "d2", "d3", "d4", "d5", "d6", "d7",
X    "a0", "a1", "a2", "a3", "a4", "a5", "a6", "sp",
X};
X
Xchar *regnm(n)
X{
X    return rnms[n];
X}
X
X#define regnm(x)  rnms[x]
X
Xchar *
Xinit_str(n)
X{
X    char *s;
X
X    switch (n) {
X    case 1:
X         s = ".dc.b";  break;
X    case 2:
X         s = ".dc.w";  break;
X    default:
X         s = ".dc.l";  break;
X    }
X    return s;
X}
X
Xtlen(n)
X{
X    switch (n) {
```

```
X    case 1:
X         return 'b';
X    case 2:
X         return 'w';
X    default:
X         return 'l';
X    }
X}
X
Xo_vinit(tp, xp)
XNODEP tp, xp;
X{
X    fprintf(output, "\t%s\t", init_str((int)tp->t_size));
X    dat_size += tp->t_size;
X
X    p2_expr(&xp);
X    asn_chk(tp, xp);
X    to_init(xp, tp);
X
X    fputc('\n', output);
X}
X
Xto_init(np, typ)
XNODEP np, typ;
X{
X    NODEP tp;
X
X    tp = allocnode();
X    tp->e_token = TCONV;
X    tp->n_tptr = typ;
X    tp->n_flags |= N_COPYT;
X    tp->n_left = np;
X    tp->e_type = E_UNARY;
X    strcpy(tp->n_name, "i cast");
X
X    genx(tp, FORINIT);
X}
X
Xout_advice(np)
XNODEP np;
X{
X    long size;
X
X    size = np->n_tptr->t_size;
X    if (size == 0)
X         return;
X
X    switch (np->e_sc) {
X    case K_AUTO:
X         fprintf(output, ";var\t%d\t%d\t", (int)size,
X              (int)np->e_offs);
X         break;
X    case K_REGISTER:
```

```
X         fprintf(output, ";reg\t%d\t%s\t", (int)size,
X              regnm(np->e_rno));
X         break;
X    default:
X         return;
X    }
X    out_nm(np);
X    fputc('\n', output);
X}
X
Xout_argreg(np)
XNODEP np;
X{
X    fprintf(output, "\tmove.%c\t%d(%s),%s\n",
X         tlen((int)np->n_tptr->t_size), (int)np->e_offs,
X         regnm(FRAMEP), regnm(np->e_rno));
X}
X
Xout_fstart(np)
XNODEP np;
X{
X    extern   int  pflag;
X
X    TO_TEXT;
X    if (np->e_sc != K_STATIC) {
X         fprintf(output, "\t.globl\t");
X         und_nnm(np);
X         fputc('\n', output);
X    }
X    und_nnm(np);
X    fprintf(output, ":\n");
X
X    if (pflag) {
X         int  tlab = new_lbl();
X
X         TO_BSS;
X         fprintf(output, "L%d:\t.ds.l\t1\n", tlab);
X         TO_TEXT;
X         fprintf(output, "\tmove.l\t#");
X         und_nnm(np);
X         fprintf(output, ",a0\n");
X
X         fprintf(output, "\tmove.l\t#L%d,a1\n", tlab);
X         fprintf(output,    "\tjsr\tmcount\n");
X    }
X}
X
Xstatic char rbuf[30];
X
Xchar *
Xregstr(regs)
X{
X    int lod, hid, loa, hia;
```

```
X    register i;
X    char *bp = rbuf;
X
X    lod = 999;
X    hid = -1;
X    for (i=DRV_START; i<=DRV_END; i++)
X         if (regs & (1<<i)) {
X              if (i < lod)  lod = i;
X              if (i > hid)  hid = i;
X         }
X    loa = 999;
X    hia = -1;
X    for (i=ARV_START; i<=ARV_END; i++)
X         if (regs & (1<<i)) {
X              if (i < loa)  loa = i;
X              if (i > hia)  hia = i;
X         }
X    if (lod < 999) {
X         if (lod != hid)
X              sprintf(bp, "%s-%s", rnms[lod], rnms[hid]);
X         else
X              sprintf(bp, "%s", rnms[lod]);
X         if (loa < 999) {
X              bp += strlen(rbuf);
X              *bp++ = '/';
X         }
X    }
X    if (loa < 999) {
X         if (loa != hia)
X              sprintf(bp, "%s-%s", rnms[loa], rnms[hia]);
X         else
X              sprintf(bp, "%s", rnms[loa]);
X    }
X    return rbuf;
X}
X
Xout_fend(regs, lsize)
Xlong lsize;
X{
X    if (lsize < 0x7fff)
X         fprintf(output, "\tlink\t%s,#-%d\n", rnms[FRAMEP], (int)lsize);
X    else
X         fprintf(output, "\tlink\t%s,#0\n\tsub.l\t#%ld,sp\n",
X              rnms[FRAMEP], lsize);
X    if (regs)
X         fprintf(output, "\tmovem.l\t%s,-(sp)\n", regstr(regs));
X}
X
Xout_fret(regs, strl)
X{
X    if (regs)
X         fprintf(output, "\tmovem.l\t(sp)+,%s\n", regstr(regs));
X    if (strl)
```

```
X        fprintf(output, "\tmove.l\t#L%d,a0\n", strl);
X    fprintf(output, "\tunlk\t%s\n\trts\n", rnms[FRAMEP]);
X}
X
Xout_fs(strl, size)
Xlong size;
X{
X    TO_BSS;
X    def_lbl(strl);
X    if (size & 1)
X        fprintf(output, "\t.ds.b\t%ld\n", size);
X    else
X        fprintf(output, "\t.ds.w\t%ld\n", size/2);
X}
X
Xout_gv(np, isbss)
Xregister NODEP np;
X{
X    long sz;
X    char c;
X
X    if (np->e_sc == K_STATIC) {
X        np->e_offs = lblnum++;
X    }
X    if (np->e_sc != K_EXTERN) {
X        to_seg(isbss ? B_SEG : D_SEG);
X        if (np->e_sc != K_STATIC) {
X            fprintf(output, "\t.globl\t");
X            out_nm(np);
X            fputc('\n', output);
X        }
X        if (isbss) {
X            if (np->e_sc == K_STATIC) {
X                out_nm(np);
X                sz = np->n_tptr->t_size;
X                c = 'b';
X                if (np->n_tptr->t_aln) {
X                    c = 'w';
X                    sz /= 2;
X                }
X                fprintf(output, ":\t.ds.%c\t%ld\n", c, sz);
X            } else {
X                fprintf(output, "\t.comm\t");
X                out_nm(np);
X                sz = np->n_tptr->t_size;
X                if (sz & 1) sz++;  /* ALCYON hack */
X                fprintf(output, ",%ld\n", sz);
X            }
X        } else {
X            out_nm(np);
X            fprintf(output, ":\n");
X        }
X    }
```

```
X}
X
Xnew_lbl()
X{
X    return lblnum++;
X}
X
Xdef_lbl(l)
X{
X    fprintf(output, "L%d:\n", l);
X}
X
Xout_br(l)
X{
X    if (l < 0)
X         error("bad branch");
X    else
X         fprintf(output, "\tbra\tL%d\n", l);
X}
X
Xstatic char *bnm[] = {
X    "",
X    "beq",
X    "bne",
X    "blt",
X    "bge",
X    "ble",
X    "bgt",
X    "bra",
X    "nop",
X    "bcs",
X    "bcc",
X    "bls",
X    "bhi"
X};
X
Xstatic char *snm[] = {
X    "",
X    "seq",
X    "sne",
X    "slt",
X    "sge",
X    "sle",
X    "sgt",
X    "",
X    "",
X    "scs",
X    "scc",
X    "sls",
X    "shi"
X};
X
Xout_b(key, l)
```

```
X{
X    if (key != B_NO)
X         fprintf(output, "\t%s\tL%d\n", bnm[key], l);
X}
X
Xout_bnol(key)
X{
X    fprintf(output, "\t%s\t", bnm[key]);
X}
X
Xout_snol(key)
X{
X    fprintf(output, "\t%s\t", snm[key]);
X}
X
Xout_d0cmp(x)
X{
X    fprintf(output, "\tcmp.w\t#%d,d0\n", x);
X}
X
Xout_d0sub(x)
X{
X    fprintf(output, "\tsub.w\t#%d,d0\n", x);
X}
X
Xout_tlbl(l)
X{
X    fprintf(output, "\t.dc.l\tL%d\n", l);
X}
X
Xout_tsw()
X{
X    fprintf(output, "\text.l\td0\n");
X    fprintf(output, "\tasl.l\t#2,d0\n");
X    fprintf(output, "\tmove.l\t4(pc,d0.l),a0\n");
X    fprintf(output, "\tjmp\t(a0)\n");
X}
X
Xout_nm(np)
XNODEP np;
X{
X    if (np->e_sc == K_STATIC)
X         fprintf(output, "L%d", (int)np->e_offs);
X    else
X         und_nnm(np);
X}
X
Xout_zi(tp)
XNODEP tp;
X{
X    char *s;
X/*
X    switch (tp->t_token) {
```

```
X    case K_FLOAT:
X         fprintf(output, "\t.float\t0.0\n");  return;
X    case K_DOUBLE:
X         fprintf(output, "\t.double\t0.0\n"); return;
X    }
X*/
X    dat_size += tp->t_size;
X    s = init_str((int)tp->t_size);
X    fprintf(output, "\t%s\t0\n", s);
X}
X
Xo_nz(sz, aln)
Xlong sz;
X{
X    dat_size += sz;
X    if (aln) {
X         if (sz & 1)
X              fprintf(output, "\t.ds.b\t1\n");
X         sz >>= 1;
X         fprintf(output, "\t.ds.w\t%ld\n", sz);
X    } else {
X         fprintf(output, "\t.ds.b\t%ld\n", sz);
X    }
X}
X
Xdumpstrs(np)
XNODEP np;
X{
X    TO_DATA;
Xmore:
X    if (np == NULL)
X         return;
X    fprintf(output, "L%d:", (int)np->g_offs);
X    out_scon(np);
X    np = np->n_next;
X    goto more;
X}
X
Xint see_esc;
X
Xout_scon(np)
XNODEP np;
X{
X    int len = 0;
X
X    if (np == NULL)
X         return 0;
X    see_esc = 0;
Xmore:
X    if (np->n_name[0]) {
X         fprintf(output, "\t.dc.b\t");
X         len += out_str(np->n_name);
X         putc('\n', output);
```

```
X    }
X    np = np->n_nmx;
X    if (np)
X         goto more;
X
X    fprintf(output, "\t.dc.b\t0\n");
X    len++;
X    dat_size += len;
X    return len;
X}
X
Xout_str(s)
Xchar *s;
X{
X    int len;
X    register c;
X
X    len = 0;
X    for ( ; c = *s; s++) {
X         if (see_esc) {     /* allow null */
X              c--;
X              see_esc = 0;
X         } else if (c == 1) {
X              see_esc = 1;
X              continue;
X         }
X         if (len) {
X              if ((len & 15) == 0)
X                   fprintf(output, "\n\t.dc.b\t");
X              else
X                   putc(',', output);
X         }
X         out_1c(c);
X         len++;
X    }
X    return len;
X}
X
Xout_asm(np)
XNODEP np;
X{
X    putc('\t', output);
Xmore:
X    fprintf(output, "%s", np->n_name);    /* no \0 or \1 please! */
X    np = np->n_nmx;
X    if (np)
X         goto more;
X    putc('\n', output);
X}
X
Xund_nnm(np)
XNODEP np;
X{
```

```
X    fputc('_', output);
X    fput_nnm(np);
X}
X
Xout_1c(c)
Xchar c;
X{
X    fprintf(output, "$%x", c & 0xff);
X}
X
Xoutcode(np)
Xregister NODEP np;
X{
X    NODEP tp;
X
X    if (np == NULL) return;
X
X    switch (np->g_type) {
X    case EV_NONE:
X         break;
X    case EV_RL:
X         outcode(np->n_right);
X         outsub(np->g_betw, np);
X         /* fall through */
X    case EV_LEFT:
X         outcode(np->n_left);
X         break;
X    case EV_LR:
X    case EV_LRSEP:
X         outcode(np->n_left);
X         outsub(np->g_betw, np);
X         /* fall through */
X    case EV_RIGHT:
X         outcode(np->n_right);
X         break;
X    default:
X         printf("bad eval %d ", np->g_type);
X    }
X    if (np->n_flags & N_COPYT)  /* g_code is a char * */
X         outsub(np->g_code, np);
X    else               /* g_code is a list of nodes */
X         for (tp=np->g_code; tp != NULL; tp = tp->g_code)
X              outsub(tp->n_name, np);
X}
X
Xoutsub(cp, np)
Xregister char *cp;
Xregister NODEP np;
X{
X    register char c;
X
X    if (cp == NULL) return;
X    while (c = *cp++)
```

```
X        if (c == '<')
X             out_let(*cp++, np->n_left);
X        else if (c == '>')
X             out_let(*cp++, np->n_right);
X        else if (c == '\'') {
X             c = *cp++;
X             fputc(c, output);
X        } else if (c == 'L')
X             seelab(*cp++, np);
X        else if (c == 'R')
X             seereg(np, *cp++);
X        else if (c >= 'A' && c <= 'Z') {
X             out_let(c, np);
X        } else
X             fputc(c, output);
X}
X
Xseereg(np, c)
XNODEP np;
X{
X    int i;
X
X    switch (c) {
X    case '0': i = np->g_rno;  break;
X    case '1': i = np->g_r1;   break;
X    case '2': i = np->g_r2;   break;
X    }
X    fprintf(output, regnm(i));
X}
X
Xout_let(c, np)
Xregister NODEP np;
X{
X    int i;
X
X    switch (c) {
X    case 'A':
X         if (np->g_flags & IMMEDID)
X              fputc('#', output);
X         out_a(np, output);
X         break;
X    case 'E':
X    case 'F': /* branch if false */
X         i = cctok(np);
X         i = (i&1) ? i+1 : i-1; /* reverse truth */
X         if (c == 'F')
X              out_bnol(i);
X         else
X              out_snol(i);
X         break;
X    case 'H': /* last a reg (for struct assign) */
X         fprintf(output, regnm(ARV_START-1));
X         break;
```

```
X    case 'K':
X         fprintf(output, "%ld", np->g_bsize);
X         break;
X    case 'N':
X         fprintf(output, "%s", np->n_name);
X         break;
X    case 'O':
X         fprintf(output, "%ld", np->g_offs);
X         break;
X    case 'Q':
X         if (np->g_flags & IMMEDID) {
X              warn("constant test expr");
X              if (np->g_token == ICON && np->g_offs == 0)
X                   fprintf(output, "\tor\t#$FF,ccr\n");
X              else
X                   fprintf(output, "\tand\t#0,ccr\n");
X              return;
X         }
X         fprintf(output, "\t%s.%c\t", isareg(np) ? "cmp" : "tst",
X              tlen(np->g_sz));
X         if (isareg(np))
X              fprintf(output, "#0,");
X         out_let('A', np);
X         fputc('\n', output);
X         break;
X    case 'S':
X         fputc(tlen(np->g_sz), output);
X         break;
X    case 'T': /* branch if true */
X         out_bnol(cctok(np));
X         break;
X    case 'U':
X         fputc(np->g_ty == ET_U ? 'u' : 's', output);
X         break;
X    case 'W': /* field width 1's */
X         fprintf(output, "$%x", ones(np->g_fldw));
X         break;
X    case 'X': /* ~(W << offset) */
X         fprintf(output, "$%x", (unsigned short)
X                   (~(ones(np->g_fldw)<<np->g_fldo)));
X         break;
X    case 'Y': /* field offset */
X         fprintf(output, "%d", np->g_fldo);
X         break;
X    case 'Z': /* field offset - 8 */
X         fprintf(output, "%d", np->g_fldo - 8);
X         break;
X    default:
X         printf("bad out_let %c ", c);
X    }
X}
X
Xout_a(np, fd)
```

```
Xregister NODEP np;
XFILE *fd;
X{
X    int offs = np->g_offs;
X
X    switch (np->g_token) {
X    case ICON:
X         fprintf(fd, "%ld", np->g_offs);
X         break;
X    case FCON:
X         /* works for ALCYON C */
X         /* otherwise depends on floating internal format */
X         fprintf(fd, "$%lx", np->g_offs);
X         break;
X    case ONAME:
X         while (np->g_flags & (CHILDNM|RCHILDNM)) {
X              np = (np->g_flags & CHILDNM) ?
X                   np->n_left : np->n_right;
X         }
X         qput_nnm(np, fd);
X         if (offs)
X              fprintf(fd, offs > 0 ? "+%d" : "%d", offs);
X         break;
X    case PUSHER:
X         fprintf(fd, "(sp)+");
X         break;
X    case OREG:
X         if (offs)
X              fprintf(fd, "%d", offs);
X         fprintf(fd, "(%s)", regnm(np->g_rno));
X         break;
X    case REGVAR:
X         fprintf(fd, regnm(np->g_rno));
X         break;
X    case ',':
X         fputc(',', fd);         /* for debug */
X         break;
X    default:
X         if (np->g_token >= BR_TOK) {
X              fprintf(fd, "B_%s", bnm[np->g_token - BR_TOK]);
X              break;
X         }
X         printf("? tok %d ", np->g_token);
X    }
X}
X
Xseelab(c, np)
Xchar c;
XNODEP np;
X{
X    c -= '1';
X    fprintf(output, "L%d", (int)np->g_bsize+c);
X}
```

```
X
Xones(n)
X{
X    return (1 << n) - 1;
X}
X
END_OF_FILE
if test 12658 -ne `wc -c <'hcc/OUT_ST.C'`; then
    echo shar: \"'hcc/OUT_ST.C'\" unpacked with wrong size!
fi
# end of 'hcc/OUT_ST.C'
fi
if test -f 'top/BRANCH.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/BRANCH.C'\"
else
echo shar: Extracting \"'top/BRANCH.C'\" \(12147 characters\)
sed "s/^X//" >'top/BRANCH.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X#include "top.h"
X
X#define  TOUCHED(bp)   ((bp)->flags & B_TOUCHED)
X
X/*
X * bcomp(bc) - return the complement of the given branch code
X *
X * Used when a branch reversal is needed.
X */
Xstatic  int
Xbcomp(bc)
Xregister int bc;
X{
X    switch (bc) {
X    case BHI: return BLS;
X    case BLS: return BHI;
X    case BCC: return BCS;
X    case BCS: return BCC;
X    case BNE: return BEQ;
X    case BEQ: return BNE;
X    case BVC: return BVS;
X    case BVS: return BVC;
X    case BPL: return BMI;
X    case BMI: return BPL;
X    case BGE: return BLT;
X    case BLT: return BGE;
```

```
X    case BGT: return BLE;
X    case BLE: return BGT;
X    default:
X         fprintf(stderr, "bcomp() - bad branch code %d\n", bc);
X         exit(1);
X    }
X}
X
X/*
X * isbranch(s) - determines if 's' is a branch opcode
X *
X * Returns 1 for branches whose destination is the first operand,
X * and 2 for branches whose dest. is the second.
X */
Xstatic  int
Xisbranch(c)
Xregister int c;
X{
X    switch (c) {
X    case BRA: case BHI: case BLS: case BCC:
X    case BCS: case BNE: case BEQ: case BVC:
X    case BVS: case BPL: case BMI: case BGE:
X    case BLT: case BGT: case BLE:
X         return 1;
X
X    case DBRA: case DBHI: case DBLS: case DBCC:
X    case DBCS: case DBNE: case DBEQ: case DBMI:
X    case DBGE: case DBLT: case DBGT: case DBLE:
X    case  DBT:
X         return 2;
X
X    default:
X         return 0;
X    }
X}
X
X/*
X * cblock(cp) - return the first block containing some code
X *
X * Starting with 'cp', find a block that has one or more instructions
X * in it. This is useful to collapse multiple null blocks into a single
X * logical point. This happens at points in the generated code where
X * there are multiple labels at the same logical location.
X */
Xstatic  BLOCK *
Xcblock(bp)
Xregister BLOCK    *bp;
X{
X    while (bp->first == NULL && bp->bcond == NULL) {
X         if (bp->bfall == NULL) {
X              fprintf(ofp, "cblock() - error in block %s\n",
X                   bp->name);
X              exit(1);
```

```
X        }
X        bp = bp->bfall;
X    }
X
X    return bp;
X}
X
X/*
X * bsplit() - split up blocks with branches inside them
X *
X * Look for branch instructions in each block. If somewhere in the middle of
X * the block, split up the block. When done, the blocks are broken down into
X * true basic blocks.
X */
Xstatic  void
Xbsplit(bp)
XBLOCK   *bp;
X{
X    register BLOCK     *cp; /* the current block */
X    register BLOCK     *np; /* new block (if needed) */
X    register INST *ip; /* current instruction */
X
X    for (cp = bp; cp != NULL ;cp = cp->chain) {
X        for (ip = cp->first; ip != NULL ;ip = ip->next) {
X            if (isbranch(ip->opcode) && ip->next != NULL) {
X                np = mksym(mktmp());
X
X                np->chain = cp->chain;
X                cp->chain = np;
X
X                np->next = cp->next;
X                cp->next = np;
X
X                np->first = ip->next;
X                np->first->prev = NULL;
X                np->last = cp->last;
X
X                cp->last = ip;
X                cp->last->next = NULL;
X
X            } else if (ip->opcode == DC) {
X                BLOCK    *db;
X
X                /*
X                 * If the instruction is part of a branch
X                 * table, both the current block and the
X                 * destination need to be marked as "reached".
X                 */
X                cp->flags |= B_ISREACHED;
X
X                if ((db = getsym(ip->src.astr)) != NULL)
X                    db->flags |= B_ISREACHED;
X                else {
```

```
X                       fprintf(stderr,
X                         "bsplit() - symbol '%s' not found\n",
X                         ip->src.astr);
X                       exit(1);
X                   }
X               }
X           }
X       }
X}
X
X/*
X * bfix() - fix up the branch pointers
X *
X * Go through each block setting up 'bcond' and 'bfall' properly. If the
X * last instruction in the block is an unconditional branch, remove it
X * and set 'bfall' instead. The idea is that there should be no branch
X * instructions left when we're done. We remember the logical effect of
X * each branch, but reconstruct the branches later in a more optimal way.
X */
Xstatic  void
Xbfix(bp)
Xregister BLOCK    *bp;
X{
X    register BLOCK     *cp; /* current block */
X    register INST *ip; /* current instruction */
X
X    for (cp = bp; cp != NULL ;cp = cp->chain) {
X
X        /* no instructions in the block */
X        if (cp->first == NULL) {
X            cp->bcond = NULL;
X            cp->bfall = cp->next;
X            continue;
X        }
X
X        /* the last instruction is a "return" */
X        if (cp->last->opcode == RTS) {
X            cp->bcond = cp->bfall = NULL;
X            cp->flags |= B_RET;
X            continue;
X        }
X
X        /* the last instruction isn't a branch */
X        if (!isbranch(cp->last->opcode)) {
X            cp->bcond = NULL;
X            cp->bfall = cp->next;
X            continue;
X        }
X
X        /*
X         * If we reach this point, then we've got a branch we need
X         * to remove at the end of this block.
X         */
```

```
X         cp->bfall = cp->next;
X         if (isbranch(cp->last->opcode) == 1) {
X              cp->bcode = cp->last->opcode;
X              cp->bcond = getsym(cp->last->src.astr);
X         } else {
X              cp->bcode = -1;
X              cp->bcond = getsym(cp->last->dst.astr);
X         }
X
X         if (cp->bcond == NULL) {
X              fprintf(stderr, "top: branch to bad label '%s'\n",
X                   cp->last->src.astr);
X              exit(1);
X         }
X
X         if (cp->bcode < 0)
X              continue;
X
X         for (ip = cp->first; ip != NULL ;ip = ip->next) {
X              if (ip->next == cp->last) {
X                   ip->next = NULL;
X                   break;
X              }
X         }
X         free(cp->last->src.astr);
X         free(cp->last);
X
X         if (cp->first == cp->last)
X              cp->first = cp->last = NULL;
X         else
X              cp->last = ip;
X         /*
X          * If the branch was unconditional, we want to represent
X          * it as a "fall through", so fix the pointers to do that.
X          */
X         if (cp->bcode == BRA) {
X              s_bdel++;
X              cp->bfall = cp->bcond;
X              cp->bcond = NULL;
X         }
X    }
X}
X
X/*
X * bclean() - remove references to empty blocks
X *
X * Called after bsplit() and bfix().
X */
Xstatic  void
Xbclean(bp)
Xregister BLOCK    *bp;
X{
X    register BLOCK     *cp;
```

```
X
X    /*
X     * First clean up references to empty blocks
X     */
X    for (cp = bp; cp != NULL ;cp = cp->chain) {
X         if (cp->bcond != NULL)
X              cp->bcond = cblock(cp->bcond);
X         if (cp->bfall != NULL)
X              cp->bfall = cblock(cp->bfall);
X    }
X
X    /*
X     * Now there are generally blocks that are still linked by the
X     * 'chain' pointers, but no longer referenced through 'bcond'
X     * or 'bfall' pointers. They don't actually need to be deleted
X     * since they won't cause trouble anywhere else.
X     */
X}
X
X#if 0
X/*
X * reachable(p1, p2) - is p2 reachable (with constraints) from p1
X *
X * Try to reach p2 from p1 by following the 'bfall' pointers, without
X * going through a block that has already been touched. We stop searching
X * after a while since this function doesn't have to be perfect, and we're
X * mainly interested in improving small loops anyway.
X */
Xstatic  bool
Xreachable(p1, p2)
Xregister BLOCK    *p1, *p2;
X{
X    register int  i;
X
X    for (i=0; (i < 40) && (p1 != NULL) ;i++, p1 = p1->bfall) {
X
X         if (TOUCHED(p1))
X              return FALSE;
X
X         if (p1 == p2)
X              return TRUE;
X    }
X    return FALSE;
X}
X#endif
X
X/*
X * bwalk() - recursive walk through the branch graph
X *
X * Starting at the entry point, walk through the block graph placing
X * blocks on the output list. By traversing the "bfall" nodes first
X * we attempt to make blocks that fall through come in order so that
X * branches are minimized.
```

```
X *
X * Joe describes this process as "grabbing the tree at the top and
X * shaking".
X *
X * The 'lp' variable below maintains our position in the generated list
X * of blocks.
X */
Xstatic  BLOCK    *lp; /* pointer to the next block in the file */
X
Xstatic  bwalk(p)
Xregister BLOCK   *p;
X{
X    if (p == NULL || TOUCHED(p))
X        return;
X
X#if 0
X    /*
X     * The following code still needs some work, so it is disabled
X     * for now...
X     */
X    /*
X     * Check to see if loop rotation is required. We "rotate" the loop
X     * by skipping the current node and traversing the 'bfall' node,
X     * and then the 'bcond' node. The function reachable() tells us
X     * that there's a loop through the 'bfall' nodes allowing us to
X     * make this optimization. The net result is that we try to move
X     * conditional branches to the bottoms of loops making the loop
X     * one instruction shorter. The overall number of branches remains
X     * the same.
X     *
X     * Rather than marking all the nodes again within the function
X     * "reachable", we just give up after a while if no loop is
X     * detected. This isn't perfect, but we get the biggest gain in
X     * small loops, and these will be detected accurately.
X     */
X    if (do_lrot && p->bcond != NULL && reachable(p->bfall, p)) {
X        s_lrot++;
X        bwalk(p->bfall);
X        bwalk(p->bcond);
X        return;
X    }
X#endif
X
X    p->flags |= B_TOUCHED;
X    lp->next = p;
X    lp = lp->next;
X
X    /*
X     * We should swap 'bfall' and 'bcond' and alter 'bcode' IF:
X     *
X     * 1. Both bfall and bcond are non-null.
X     * 2. The branch is a simple one (not a DBcc inst.)
X     * 3. Branch reversals are enabled.
```

```
X     * 4. The block at bfall has already been touched.
X     * 5. The block at bcond hasn't been touched.
X     *
X     * In this case, we can avoid an extra branch if we reverse the
X     * sense of the conditional branch, and swap the pointers.
X     */
X    if (p->bfall != NULL && p->bcond != NULL && p->bcode >= 0 &&
X        do_brev && TOUCHED(p->bfall) && !TOUCHED(p->bcond)) {
X         register BLOCK     *tmp;
X
X         s_brev++;
X         tmp = p->bfall;
X         p->bfall = p->bcond;
X         p->bcond = tmp;
X         if ((p->bcode = bcomp(p->bcode)) < 0) {
X              fprintf(stderr, "top: internal error in bwalk()\n");
X              exit(1);
X         }
X    }
X    bwalk(p->bfall);
X    bwalk(p->bcond);
X}
X
X/*
X * bsort() - branch optimization
X *
X * Initialize and terminate the 'next' pointers before and after
X * traversing the branch graph.
X */
Xstatic  void
Xbsort(bp)
Xregister BLOCK    *bp;
X{
X    register BLOCK     *cb;
X
X    lp = bp;
X
X    bwalk(bp);    /* traverse the tree starting at the entry point */
X
X    /*
X     * Now look for other parts of the tree that don't appear to be
X     * reachable, but really are. This can happen through the jump
X     * tables created for switch statements. For each such block,
X     * walk the subtree starting with it.
X     */
X    for (cb = bp; cb != NULL ;cb = cb->chain) {
X         if (!TOUCHED(cb) && (cb->flags & B_ISREACHED))
X              bwalk(cb);
X    }
X
X    lp->next = NULL;
X}
X
```

```
X/*
X * bsetlab() - figure out which blocks really need labels
X *
X * This can be called AFTER bsort() to mark the blocks that are going to
X * need labels. Anything that we're going to branch to later gets a label.
X */
Xstatic  void
Xbsetlab(bp)
Xregister BLOCK    *bp;
X{
X    for (; bp != NULL ;bp = bp->next) {
X         if (bp->bcond != NULL)
X              cblock(bp->bcond)->flags |= B_LABEL;
X         if (bp->bfall != NULL && bp->bfall != bp->next)
X              cblock(bp->bfall)->flags |= B_LABEL;
X         /*
X          * If the block is reached via a jump table, then we
X          * need a label on THIS block directly.
X          */
X         if (bp->flags & B_ISREACHED)
X              bp->flags |= B_LABEL;
X    }
X}
X
X/*
X * bjoin() - join blocks that always come together
X *
X * If block 'b' always follows block 'a' unconditionally, then move the
X * instructions in 'b' to the end of 'a', and remove 'b'. This allows us
X * to do better peephole optimization since we can optimize sequences that
X * used to span blocks.
X */
Xbjoin(bp)
Xregister BLOCK    *bp;
X{
X    BLOCK     *np, *bnext;
X
X    for (; bp != NULL ;bp = bnext) {
X         bnext = bp->next;
X
X         /*
X          * First block can't end with a conditional branch.
X          */
X         if (bp->bcond != NULL)
X              continue;
X
X         /*
X          * Block must fall through to the next thing in the file.
X          */
X         if (bp->next == NULL || bp->next != bp->bfall)
X              continue;
X
X         np = bp->next;
```

```
X
X        /*
X         * Second block can't be the destination of a branch.
X         */
X        if (np->flags & B_LABEL)
X            continue;
X
X        /*
X         * Join the blocks...
X         */
X        if (np->first != NULL)
X            np->first->prev = bp->last;
X
X        if (bp->last != NULL)
X            bp->last->next = np->first;
X        else {
X            bp->first = np->first;
X            bp->last = np->last;
X        }
X
X        if (np->flags & B_RET)
X            bp->flags |= B_RET;
X
X        bp->last = np->last;
X        bp->bfall = np->bfall;
X        bp->bcond = np->bcond;
X        bp->bcode = np->bcode;
X        bp->next = np->next;
X
X        /*
X         * Fix pointers so we don't free the instructions
X         * twice later on.
X         */
X        np->first = NULL;
X        np->last = NULL;
X
X        /*
X         * If we've done a join, then we want to loop again on
X         * the current block to see if any others can be joined.
X         */
X        bnext = bp;
X    }
X}
X
X/*
X * bopt() - coordinates branch optimization for the given function
X */
Xvoid
Xbopt(bp)
Xregister BLOCK    *bp;
X{
X    bsplit(bp);
X    bfix(bp);
```

```
X    bclean(bp);
X    bsort(bp);
X    bsetlab(bp);
X    bjoin(bp);
X}
END_OF_FILE
if test 12147 -ne `wc -c <'top/BRANCH.C'`; then
    echo shar: \"'top/BRANCH.C'\" unpacked with wrong size!
fi
# end of 'top/BRANCH.C'
fi
if test -f 'top/PEEP3.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/PEEP3.C'\"
else
echo shar: Extracting \"'top/PEEP3.C'\" \(12352 characters\)
sed "s/^X//" >'top/PEEP3.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X
X/*
X * 3-instruction peephole optimizations
X */
X
X#include "top.h"
X
X
X/*
X * ipeep3(bp, ip) - look for 3-instruction optimizations at the given inst.
X */
Xstatic bool
Xipeep3(bp, i1)
Xregister BLOCK    *bp;
Xregister INST     *i1;
X{
X    register INST *i2 = i1->next;    /* the next instruction */
X    INST *i3 = i1->next->next;  /* the third instruction */
X
X    register int  op1 = i1->opcode;
X    register int  op2 = i2->opcode;
X    register int  op3 = i3->opcode;
X
X    /*
X     *   move.l    Am, Dn        =>   lea  N(Am), Ao
X     *   add.l     #N, Dn
X     *   move.l    Dn, Ao
```

```
X     *
X     *   Also, Dn must be dead after the third instruction.
X     */
X    if ((op1 == MOVE) && (i1->flags & LENL) &&
X        (i1->src.amode == REG) &&
X        ISA(i1->src.areg) &&
X        (i1->dst.amode == REG) &&
X        ISD(i1->dst.areg)) {
X
X         if (((op2 == ADD) || (op2 == ADDQ)) &&
X             (i2->flags & LENL) &&
X             (i2->src.amode == IMM) &&
X             DOK(i2->src.disp) &&
X             (i2->dst.amode == REG) &&
X             (i2->dst.areg == i1->dst.areg)) {
X
X              if ((op3 == MOVE) && (i3->flags & LENL) &&
X                  (i3->src.amode == REG) &&
X                  (i3->src.areg == i1->dst.areg) &&
X                  (i3->dst.amode == REG) &&
X                  ISA(i3->dst.areg) &&
X                  ((i3->live & RM(i3->src.areg)) == 0)) {
X
X                   /*
X                    * rewrite i1 and delete i2 and i3
X                    */
X                   i1->opcode = LEA;
X                   i1->flags = 0;
X                   i1->dst = i3->dst;
X
X                   i1->src.amode = REGID;
X                   i1->src.disp = i2->src.disp;
X
X                   delinst(bp, i2);
X                   delinst(bp, i3);
X                   DBG(printf("%d ", __LINE__))
X                   return TRUE;
X              }
X         }
X    }
X
X    /*
X     *   move.l    Dm, Dn        =>   move.l    Dm, Ao
X     *   add.l     #N, Dn             lea  N(Ao), Ao
X     *   move.l    Dn, Ao
X     *
X     *   Also, Dn must be dead after the third instruction.
X     */
X    if ((op1 == MOVE) && (i1->flags & LENL) &&
X        (i1->src.amode == REG) &&
X        ISD(i1->src.areg) &&
X        (i1->dst.amode == REG) &&
X        ISD(i1->dst.areg)) {
```

```
X
X        if (((op2 == ADD) || (op2 == ADDQ)) &&
X            (i2->flags & LENL) &&
X            (i2->src.amode == IMM) &&
X            DOK(i2->src.disp) &&
X            (i2->dst.amode == REG) &&
X            (i2->dst.areg == i1->dst.areg)) {
X
X             if ((op3 == MOVE) && (i3->flags & LENL) &&
X                 (i3->src.amode == REG) &&
X                 (i3->src.areg == i1->dst.areg) &&
X                 (i3->dst.amode == REG) &&
X                 ISA(i3->dst.areg) &&
X                 ((i3->live & RM(i3->src.areg)) == 0)) {
X
X                  /*
X                   * rewrite i1 and i2 and delete i3
X                   */
X                  i1->dst.areg = i3->dst.areg;
X
X                  i2->opcode = LEA;
X                  i2->flags = 0;
X                  i2->dst = i3->dst;
X
X                  i2->src.amode = REGID;
X                  i2->src.areg = i2->dst.areg;
X                  i2->src.disp = i2->src.disp;
X
X                  delinst(bp, i3);
X                  DBG(printf("%d ", __LINE__))
X                  return TRUE;
X             }
X        }
X    }
X
X    /*
X     *   move.l    Am, Dn        =>   lea  -N(Am), Ao
X     *   sub.l     #N, Dn
X     *   move.l    Dn, Ao
X     *
X     *   Also, Dn must be dead after the third instruction.
X     */
X    if ((op1 == MOVE) && (i1->flags & LENL) &&
X        (i1->src.amode == REG) &&
X        ISA(i1->src.areg) &&
X        (i1->dst.amode == REG) &&
X        ISD(i1->dst.areg)) {
X
X        if (((op2 == SUB) || (op2 == ADDQ)) &&
X            (i2->flags & LENL) &&
X            (i2->src.amode == IMM) &&
X            DOK(i2->src.disp) &&
X            (i2->dst.amode == REG) &&
```

```
X            (i2->dst.areg == i1->dst.areg)) {
X
X             if ((op3 == MOVE) && (i3->flags & LENL) &&
X                 (i3->src.amode == REG) &&
X                 (i3->src.areg == i1->dst.areg) &&
X                 (i3->dst.amode == REG) &&
X                 ISA(i3->dst.areg) &&
X                 ((i3->live & RM(i3->src.areg)) == 0)) {
X
X                  /*
X                   * rewrite i1 and delete i2 and i3
X                   */
X                  i1->opcode = LEA;
X                  i1->flags = 0;
X                  i1->dst = i3->dst;
X
X                  i1->src.amode = REGID;
X                  i1->src.disp = -i2->src.disp;
X
X                  delinst(bp, i2);
X                  delinst(bp, i3);
X                  DBG(printf("%d ", __LINE__))
X                  return TRUE;
X             }
X        }
X    }
X
X    /*
X     *   move.l    Am, Dn        =>   lea  0(Am, Do), Ap
X     *   add.x     Do, Dn
X     *   move.l    Dn, Ap
X     *
X     *   The second instruction can be either a word or long add.
X     *   Also, Dn must be dead after the third instruction.
X     */
X    if ((op1 == MOVE) && (i1->flags & LENL) &&
X        (i1->src.amode == REG) &&
X        ISA(i1->src.areg) &&
X        (i1->dst.amode == REG) &&
X        ISD(i1->dst.areg)) {
X
X         if (((op2 == ADD) || (op2 == ADDQ)) &&
X             (i2->flags & (LENL|LENW)) &&
X             (i2->src.amode == REG) &&
X             ISD(i2->src.areg) && (i1->dst.areg != i2->src.areg) &&
X             (i2->dst.amode == REG) &&
X             (i2->dst.areg == i1->dst.areg)) {
X
X              if ((op3 == MOVE) && (i3->flags & LENL) &&
X                  (i3->src.amode == REG) &&
X                  (i3->src.areg == i1->dst.areg) &&
X                  (i3->dst.amode == REG) &&
X                  ISA(i3->dst.areg) &&
```

```
X                ((i3->live & RM(i3->src.areg)) == 0)) {
X
X                 /*
X                  * rewrite i1 and delete i2 and i3
X                  */
X                 i1->opcode = LEA;
X                 i1->flags = 0;
X                 i1->dst = i3->dst;
X
X                 i1->src.amode = REGIDX;
X                 if (i2->flags & LENL)
X                      i1->src.amode |= XLONG;
X                 i1->src.ireg = i2->src.areg;
X                 i1->src.disp = 0;
X
X                 delinst(bp, i2);
X                 delinst(bp, i3);
X                 DBG(printf("%d ", __LINE__))
X                 return TRUE;
X             }
X         }
X    }
X
X    /*
X     *   move.l    X(Am), Dn =>   move.l    X(Am), Ao
X     *   add.l     #N, Dn
X     *   move.l    Dn, Ao             lea  N(Ao), Ao
X     *
X     *   Also, Dn must be dead after the third instruction.
X     */
X    if ((op1 == MOVE) && (i1->flags & LENL) &&
X        (i1->src.amode == REGI || i1->src.amode == REGID) &&
X        (i1->dst.amode == REG) &&
X        ISD(i1->dst.areg)) {
X
X         if (((op2 == ADD) || (op2 == ADDQ)) &&
X             (i2->flags & LENL) &&
X             (i2->src.amode == IMM) &&
X             DOK(i2->src.disp) &&
X             (i2->dst.amode == REG) &&
X             (i2->dst.areg == i1->dst.areg)) {
X
X              if ((op3 == MOVE) && (i3->flags & LENL) &&
X                  (i3->src.amode == REG) &&
X                  (i3->src.areg == i1->dst.areg) &&
X                  (i3->dst.amode == REG) &&
X                  ISA(i3->dst.areg) &&
X                  ((i3->live & RM(i3->src.areg)) == 0)) {
X
X                  /*
X                   * rewrite i1 and i3 and delete i2
X                   */
X                  i1->dst = i3->dst;
```

```
X
X                i3->opcode = LEA;
X                i3->flags = 0;
X                i3->src.amode = REGID;
X                i3->src.areg = i3->dst.areg;
X                i3->src.disp = i2->src.disp;
X
X                delinst(bp, i2);
X                DBG(printf("%d ", __LINE__))
X                return TRUE;
X            }
X        }
X    }
X
X    /*
X     *   move.x    X, Dn         =>   move.x    X, Do
X     *   ext.y     Dn            ext.y     Do
X     *   move.y    Dn, Do
X     *
X     *   Where Dn is dead.
X     */
X    if ((op1 == MOVE)&&(op2 == EXT)&&(op3 == MOVE)&&
X        (i1->dst.amode == REG) && ISD(i1->dst.areg) &&
X        (i2->src.amode == REG) && (i3->src.amode == REG) &&
X        (i3->dst.amode == REG) && ISD(i3->dst.areg) &&
X        (i1->dst.areg == i2->src.areg) && (i1->dst.areg == i3->src.areg) &&
X        (i2->flags == i3->flags)) {
X
X        if ((i3->live & RM(i3->src.areg)) == 0) {
X            i1->dst.areg = i3->dst.areg;
X            i2->src.areg = i3->dst.areg;
X
X            delinst(bp, i3);
X            DBG(printf("%d ", __LINE__))
X            return TRUE;
X        }
X    }
X
X    /*
X     *   move.l    X, Dm         =>   move.l    X, An
X     *   INST              INST
X     *   move.l    Dm, An            ...deleted...
X     *
X     *   where INST doesn't modify Dm, and Dm is dead after i3
X     */
X    if ((op1 == MOVE) && (op3 == MOVE) &&
X        (i1->dst.amode == REG) && ISD(i1->dst.areg) &&
X        (i3->src.amode == REG) && (i1->dst.areg == i3->src.areg) &&
X        (i3->dst.amode == REG) && ISA(i3->dst.areg) &&
X        !sets(i3, RM(i3->src.areg))) {
X
X        if ((i3->live & i3->src.areg) == 0) {
X            i1->dst.areg = i3->dst.areg;
```

```
X             delinst(bp, i3);
X
X             DBG(printf("%d ", __LINE__))
X             return TRUE;
X         }
X    }
X
X    /*
X     *   move.l    Am, An             ...deleted...
X     *   addq.l    #1, Am             ...deleted...
X     *   ... stuff ...           ... stuff ...
X     *   ???.b     ..(An)..  =>   ???.b     ..(Am)+..
X     *
X     *   An must be dead after the last instruction. Nothing in
X     *   "stuff" can modify Am.
X     */
X    if ((op1 == MOVE) && (i1->flags & LENL) &&
X        (i1->src.amode == REG) && ISA(i1->src.areg) &&
X        (i1->dst.amode == REG) && ISA(i1->dst.areg)) {
X
X         int  rm = i1->src.areg;
X         int  rn = i1->dst.areg;
X
X         if (((op2 == ADD) || (op2 == ADDQ)) &&
X             (i2->flags & LENL) &&
X             (i2->src.amode == IMM) && (i2->src.disp == 1) &&
X             (i2->dst.amode == REG) &&
X             (i2->dst.areg == rm)) {
X
X             while (i3 != NULL) {
X                  if (sets(i3, RM(rm)))
X                       goto end7;
X
X                  if (i3->src.amode==REGI && i3->src.areg==rn) {
X                       if (i3->live & RM(rn))
X                            goto end7;
X
X                       if ((i3->flags & LENB) == 0)
X                            goto end7;
X
X                       i3->src.amode |= INC;
X                       i3->src.areg = rm;
X
X                       delinst(bp, i1);
X                       delinst(bp, i2);
X                       DBG(printf("%d ", __LINE__))
X                       return TRUE;
X                  }
X                  if (i3->dst.amode==REGI && i3->dst.areg==rn) {
X                       if (i3->live & RM(rn))
X                            goto end7;
X
X                       if ((i3->flags & LENB) == 0)
```

```
X                         goto end7;
X
X                    i3->dst.amode |= INC;
X                    i3->dst.areg = rm;
X
X                    delinst(bp, i1);
X                    delinst(bp, i2);
X                    DBG(printf("%d ", __LINE__))
X                    return TRUE;
X               }
X
X               if (i3->next == NULL)
X                    goto end7;
X               else
X                    i3 = i3->next;
X
X           }
X       }
X   }
Xend7:
X
X   /*
X    *   move.l    Am, An
X    *   addq.l    #2, Am
X    *   ... stuff ...
X    *   ???.w     ..(An).. =>   ???.w     ..(Am)+..
X    *
X    *   An must be dead after the last instruction. Nothing in
X    *   "stuff" can modify Am.
X    */
X   if ((op1 == MOVE) && (i1->flags & LENL) &&
X       (i1->src.amode == REG) && ISA(i1->src.areg) &&
X       (i1->dst.amode == REG) && ISA(i1->dst.areg)) {
X
X       int  rm = i1->src.areg;
X       int  rn = i1->dst.areg;
X
X       if (((op2 == ADD) || (op2 == ADDQ)) &&
X           (i2->flags & LENL) &&
X           (i2->src.amode == IMM) && (i2->src.disp == 2) &&
X           (i2->dst.amode == REG) &&
X           (i2->dst.areg == rm)) {
X
X            while (i3 != NULL) {
X                 if (sets(i3, RM(rm)))
X                      goto end9;
X
X                 if (i3->src.amode==REGI && i3->src.areg==rn) {
X                      if (i3->live & RM(rn))
X                           goto end9;
X
X                      if ((i3->flags & LENW) == 0)
X                           goto end9;
```

```
X
X                    i3->src.amode |= INC;
X                    i3->src.areg = rm;
X
X                    delinst(bp, i1);
X                    delinst(bp, i2);
X                    DBG(printf("%d ", __LINE__))
X                    return TRUE;
X               }
X               if (i3->dst.amode==REGI && i3->dst.areg==rn) {
X                    if (i3->live & RM(rn))
X                         goto end9;
X
X                    if ((i3->flags & LENW) == 0)
X                         goto end9;
X
X                    i3->dst.amode |= INC;
X                    i3->dst.areg = rm;
X
X                    delinst(bp, i1);
X                    delinst(bp, i2);
X                    DBG(printf("%d ", __LINE__))
X                    return TRUE;
X               }
X
X               if (i3->next == NULL)
X                    goto end9;
X               else
X                    i3 = i3->next;
X
X             }
X        }
X   }
Xend9:
X
X   /*
X    *   move.l    Am, An
X    *   addq.l    #4, Am
X    *   ... stuff ...
X    *   ???.l     ..(An)..  =>   ???.l     ..(Am)+..
X    *
X    *   An must be dead after the last instruction. Nothing in
X    *   "stuff" can modify Am.
X    */
X   if ((op1 == MOVE) && (i1->flags & LENL) &&
X       (i1->src.amode == REG) && ISA(i1->src.areg) &&
X       (i1->dst.amode == REG) && ISA(i1->dst.areg)) {
X
X        int  rm = i1->src.areg;
X        int  rn = i1->dst.areg;
X
X        if (((op2 == ADD) || (op2 == ADDQ)) &&
X            (i2->flags & LENL) &&
```

```
X             (i2->src.amode == IMM) && (i2->src.disp == 4) &&
X             (i2->dst.amode == REG) &&
X             (i2->dst.areg == rm)) {
X
X             while (i3 != NULL) {
X                  if (sets(i3, RM(rm)))
X                       goto end11;
X
X                  if (i3->src.amode==REGI && i3->src.areg==rn) {
X                       if (i3->live & RM(rn))
X                            goto end11;
X
X                       if ((i3->flags & LENL) == 0)
X                            goto end11;
X
X                       i3->src.amode |= INC;
X                       i3->src.areg = rm;
X
X                       delinst(bp, i1);
X                       delinst(bp, i2);
X                       DBG(printf("%d ", __LINE__))
X                       return TRUE;
X                  }
X                  if (i3->dst.amode==REGI && i3->dst.areg==rn) {
X                       if (i3->live & RM(rn))
X                            goto end11;
X
X                       if ((i3->flags & LENL) == 0)
X                            goto end11;
X
X                       i3->dst.amode |= INC;
X                       i3->dst.areg = rm;
X
X                       delinst(bp, i1);
X                       delinst(bp, i2);
X                       DBG(printf("%d ", __LINE__))
X                       return TRUE;
X                  }
X
X                  if (i3->next == NULL)
X                       goto end11;
X                  else
X                       i3 = i3->next;
X
X             }
X        }
X    }
Xend11:
X
X    return FALSE;
X}
X
X/*
```

```
X * peep3(bp) - scan blocks starting at 'bp'
X */
Xbool
Xpeep3(bp)
Xregister BLOCK    *bp;
X{
X    register INST *ip;
X    register bool changed = FALSE;
X
X    DBG(printf("p3: "))
X    for (; bp != NULL ;bp = bp->next) {
X         ip = bp->first;
X         while (ip!=NULL && ip->next != NULL && ip->next->next != NULL) {
X              if (ipeep3(bp, ip)) {
X                   s_peep3++;
X                   bprep(bp);
X                   changed = TRUE;
X                   /*
X                    * If we had a match, then any instruction
X                    * could have been deleted, so the safe thing
X                    * to do is to go to the next block.
X                    */
X                   break;
X              } else
X                   ip = ip->next;
X         }
X    }
X    DBG(printf("\n"); fflush(stdout))
X    return changed;
X}
END_OF_FILE
if test 12352 -ne `wc -c <'top/PEEP3.C'`; then
    echo shar: \"'top/PEEP3.C'\" unpacked with wrong size!
fi
# end of 'top/PEEP3.C'
fi
echo shar: End of archive 6 \(of 9\).
cp /dev/null ark6isdone
MISSING=""
for I in 1 2 3 4 5 6 7 8 9 ; do
    if test ! -f ark${I}isdone ; then
     MISSING="${MISSING} ${I}"
    fi
done
if test "${MISSING}" = "" ; then
    echo You have unpacked all 9 archives.
    rm -f ark[1-9]isdone ark[1-9][0-9]isdone
else
    echo You still need to unpack the following archives:
    echo "        " ${MISSING}
fi
##  End of shell archive.
exit 0
```

From 6404382602428685687
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,7a0b4c902c3453fb,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-25 10:15:31 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!ames!sun-barr!newstop!sun!
sunfedcomm!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i098: sozobon1.2 -- Update to Sozobon C compiler part07/09
Keywords: shar, C
Message-ID: <34611@grapevine.uucp>
Date: 25 Oct 89 17:15:31 GMT
Sender: daemon@sunfedcomm
Reply-To: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Lines: 2836
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Posting-number: Volume 2, Issue 98
Archive-name: sozobon1.2/part07

```
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of archive 7 (of 9)."
# Contents:  hcc/GUNK.C hcc/P2.C hcc/PRE.C
# Wrapped by koreth@panarthea on Tue Oct 24 18:40:46 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'hcc/GUNK.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/GUNK.C'\"
else
echo shar: Extracting \"'hcc/GUNK.C'\" \(15002 characters\)
sed "s/^X//" >'hcc/GUNK.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X * 
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  gunk.c
X *
X *  Transformations on expression trees
X *  Most of this stuff is because we cant handle
```

```
X *  floats, long mul/div, or fields directly.
X */
X
X#include <stdio.h>
X#include "param.h"
X#include "bstok.h"
X#include "tytok.h"
X#include "flags.h"
X#include "nodes.h"
X#include "gen.h"
X
XNODEP copyone();
X
X#define gwiden(x) ((x)==1 ? 2 : (x))
X#define isfield(np)   ((np)->g_token == '.' && (np)->g_fldw)
X
XNODEP npar1, npar2, npar3;
Xchar *spar1, *spar2, *spar3;
Xint ipar1, ipar2, ipar3;
X
Xstruct rule {
X    int (*match)();        /* test for transformation needed */
X    int (*rewri)();        /* rewrite function */
X};
X
Xint m_unfold(), unfold(), m_cast(), cast(), m_inline(), inline();
Xint m_hardas(), hardas(), m_fcmp(), fcmp(), m_md_shf(), md_shf();
Xint m_eident(), eident(), m_incdec(), incdec(), m_fldas(), fldas();
X
Xstruct rule gunktbl[] = {
X    {m_unfold, unfold},
X    {m_cast, cast},
X    {m_md_shf, md_shf},
X    {m_eident, eident},
X    {m_incdec, incdec},
X    {m_hardas, hardas},
X    {m_inline, inline}, /* must cast before inline */
X    {m_fcmp, fcmp},
X    {m_fldas, fldas},
X    {0}
X};
X
Xint anygunk;
X
Xgunk(np)
XNODEP np;
X{
X    do {
X        anygunk = 0;
X        gunks(np);
X    } while (anygunk);
X}
X
```

```
Xgunks(np)
Xregister NODEP np;
X{
X    switch (np->g_type) {
X    case E_BIN:
X         gunks(np->n_right);
X    case E_UNARY:
X         gunks(np->n_left);
X    }
X    gunk1(np);
X}
X
Xgunk1(np)
XNODEP np;
X{
X    register struct rule *p;
X
X    for (p=gunktbl; p->match; p++)
X         if ((*p->match)(np)) {
X              anygunk++;
X              (*p->rewri)(np);
X              return;
X         }
X}
X
X/*
X *  Change pointer arithmetic to equivalent trees
X *       (main thing is to mult or div by object size)
X */
Xm_unfold(np)
XNODEP np;
X{
X    switch (np->g_token) {
X    case PTRADD:
X         ipar1 = '+';
X         return 1;
X    case PTRSUB:
X         ipar1 = '-';
X         return 1;
X    case PTRDIFF:
X         ipar1 = 0;
X         return 1;
X    case ASSIGN PTRADD:
X         ipar1 = ASSIGN '+';
X         return 1;
X    case ASSIGN PTRSUB:
X         ipar1 = ASSIGN '-';
X         return 1;
X    }
X    return 0;
X}
X
Xunfold(np)
```

```
XNODEP np;
X{
X    if (ipar1) {
X         ins_mul(np, np->g_offs);
X         np->g_token = ipar1;
X    } else {
X         ins_div(np, np->g_offs);
X    }
X}
X
XNODEP
Xnewgcon(kon, ty, sz)
Xlong kon;
X{
X    register NODEP kp;
X
X    kp = allocnode();
X    kp->g_token = ICON;
X    sprintf(kp->n_name, "%ld", kon);
X    kp->g_offs = kon;
X    kp->g_type = E_LEAF;
X    kp->g_ty = ty;
X    kp->g_sz = sz;
X    return kp;
X}
X
Xins_mul(np, kon)
XNODEP np;
Xlong kon;
X{
X    NODEP rp = np->n_right;
X    register NODEP mp, kp;
X
X    if (kon == 1)
X         return;
X    if (rp->g_token == ICON) {
X         rp->g_offs *= kon;
X         rp->g_sz = gwiden(rp->g_sz);
X         return;
X    }
X
X    mp = allocnode();
X    mp->g_token = '*';
X    sprintf(mp->n_name, "p*");
X    mp->g_type = E_BIN;
X    mp->g_ty = rp->g_ty;
X    mp->g_sz = gwiden(rp->g_sz);
X
X    kp = newgcon(kon, mp->g_ty, mp->g_sz);
X
X    mp->n_right = kp;
X    mp->n_left = np->n_right;
X    np->n_right = mp;
```

```
X}
X
Xins_div(np, kon)
Xregister NODEP np;
Xlong kon;
X{
X    register NODEP tp, kp;
X
X    kp = newgcon(kon, np->g_ty, np->g_sz);
X
X    tp = copyone(np);
X    tp->g_token = '-';
X    tp->n_left = np->n_left;
X    tp->n_right = np->n_right;
X    tp->g_sz = SIZE_P;
X    tp->g_ty = ET_U;
X
X    np->n_left = tp;
X    np->n_right = kp;
X    np->g_type = E_BIN;
X    np->g_token = '/';
X    sprintf(np->n_name, "p/");
X}
X
X#define CAST_LN   1
X#define CAST_RN 2
X#define CAST_LLONG    3
X
X/*
X *  Insert needed (implied) casts
X */
Xm_cast(np)
XNODEP np;
X{
X    NODEP lp = np->n_left;
X
X    switch (np->g_type) {
X    case E_LEAF:
X         return 0;
X    case E_BIN:
X         return bm_cast(np);
X    }
X    /* must be unary */
X    switch (np->g_token) {
X    case UNARY '-':
X    case '~':
X         return castup(lp, np, CAST_LN);
X    case TCONV:
X         return fcastlong(np);
X    }
X    return 0;
X}
X
```

```
Xbm_cast(np)
Xregister NODEP np;
X{
X    NODEP lp = np->n_left, rp = np->n_right;
X
X    if (isassign(np->g_token)) {
X         if (castup(rp, lp, CAST_RN))
X              return 1;
X         if (castmagic(rp, lp, CAST_RN, np->g_token - (ASSIGN 0)))
X              return 1;
X         return 0;
X    }
X
X    switch (np->g_token) {
X    case '=':
X         if (np->g_ty == ET_A)
X              return 0;
X         return castany(rp, lp, CAST_RN);
X
X    case '<':
X    case '>':
X    case DOUBLE '=':
X    case NOTEQ:
X    case LTEQ:
X    case GTEQ:
X         if (castup(lp, rp, CAST_LN))
X              return 1;
X         return castup(rp, lp, CAST_RN);
X
X    case '(':
X    case ',':
X    case '?':
X    case DOUBLE '&':
X    case DOUBLE '|':
X         return 0;
X
X    case DOUBLE '<':
X    case DOUBLE '>':
X         if (castup(lp, np, CAST_LN))
X              return 1;
X         return castany(rp, np, CAST_RN);
X
X    default:
X         if (castup(lp, np, CAST_LN))
X              return 1;
X         return castup(rp, np, CAST_RN);
X    }
X    return 0;
X}
X
Xfcastlong(np)
XNODEP np;
X{
```

```
X    NODEP lp = np->n_left;
X
X    if (red_con(lp))
X         return 0;
X    if (np->g_ty == ET_F && lp->g_ty != ET_F && lp->g_sz != SIZE_L) {
X         ipar1 = CAST_LLONG;
X         return 1;
X    }
X    if (lp->g_ty == ET_F && np->g_ty != ET_F && np->g_sz != SIZE_L) {
X         ipar1 = CAST_LLONG;
X         return 1;
X    }
X    return 0;
X}
X
Xcastup(lowp, hip, par)
XNODEP lowp, hip;
X{
X    if (stronger(hip, lowp)) {
X         ipar1 = par;
X         npar1 = hip;
X         return 1;
X    }
X    return 0;
X}
X
Xcastmagic(p1, p2, par, tok)
XNODEP p1, p2;
X{
X    if (xstronger(p1,p2) && magicop(tok)) {
X         ipar1 = par;
X         npar1 = p2;
X         return 1;
X    }
X    return 0;
X}
X
Xcastany(p1, p2, par)
XNODEP p1, p2;
X{
X    if (p1->g_sz != p2->g_sz ||
X         ((p1->g_ty == ET_F) != (p2->g_ty == ET_F))) {
X         ipar1 = par;
X         npar1 = p2;
X         return 1;
X    }
X    return 0;
X}
X
Xcast(np)
XNODEP np;
X{
X    switch (ipar1) {
```

```
X    case CAST_LN:
X         castsub(npar1->g_ty, npar1->g_sz, &np->n_left, np->n_left);
X         break;
X    case CAST_RN:
X         castsub(npar1->g_ty, npar1->g_sz, &np->n_right, np->n_right);
X         break;
X    case CAST_LLONG:
X         castsub(ET_S, SIZE_L, &np->n_left, np->n_left);
X         break;
X    }
X}
X
Xcastsub(ty, sz, npp, np)
XNODEP *npp, np;
X{
X    register NODEP tp;
X
X    /* ICON cast optimization */
X    if (np->g_token == ICON &&
X        np->g_ty == ty &&
X        np->g_sz < sz) {
X         np->g_sz = sz;
X         return;
X    }
X
X    tp = allocnode();
X    tp->g_token = TCONV;
X    strcpy(tp->n_name, "cast up");
X    tp->n_left = np;
X    *npp = tp;
X    tp->g_sz = sz;
X    tp->g_ty = ty;
X    tp->g_type = E_UNARY;
X}
X
X/*
X * Change stuff computer cant do to calls to inline functions
X * (in this case, all floats and long *%/)
X */
Xm_inline(np)
XNODEP np;
X{
X    int isfloat, isuns;
X
X    if (np->g_type == E_LEAF)
X         return 0;
X
X    if (np->g_ty == ET_A)
X         return 0;
X    isfloat = (np->g_ty == ET_F);
X    isuns = (np->g_ty == ET_U);
X
X    if (np->g_type == E_UNARY) {
```

```
X         switch (np->g_token) {
X         case UNARY '-':
X              if (!isfloat) return 0;
X              spar1 = "%fpneg";
X              return 1;
X         case TCONV:
X              if ((np->n_left->g_ty == ET_F) == isfloat)
X                   return 0;
X              if (red_con(np->n_left))
X                   return 0;
X              spar1 = isfloat ? "fpltof" : "fpftol";
X              return 1;
X         }
X         return 0;
X    }
X
X    if (np->g_sz != 4) /* longs or floats only */
X         return 0;
X
X    switch (np->g_token) {
X    case '*':
X         spar1 = isfloat ? "%fpmul" : (isuns ? "%lmulu" : "%lmul");
X         return 1;
X    case '/':
X         spar1 = isfloat ? "%fpdiv" : (isuns ? "%ldivu" : "%ldiv");
X         return 1;
X    case '+':
X         if (!isfloat) return 0;
X         spar1 = "%fpadd";
X         return 1;
X    case '-':
X         if (!isfloat) return 0;
X         spar1 = "%fpsub";
X         return 1;
X    case '%':
X         spar1 = isuns ? "%lremu" : "%lrem";
X         return 1;
X    }
X    return 0;
X}
X
Xinline(np)
XNODEP np;
X{
X    register NODEP nmp, cmap;
X    int isunary;
X
X    isunary = (np->g_type == E_UNARY);
X
X    if (isunary) {
X         np->n_right = np->n_left;
X         np->g_type = E_BIN;
X    } else {
```

```
X         cmap = copyone(np);
X         cmap->n_left = np->n_left;
X         cmap->n_right = np->n_right;
X         np->n_right = cmap;
X
X         cmap->g_token = ',';
X         cmap->g_offs = 2;
X         strcpy(cmap->n_name, ",inl");
X    }
X
X    nmp = allocnode();
X    np->n_left = nmp;
X
X    np->g_token = '(';
X    strcpy(np->n_name, "inline");
X
X    nmp->g_token = ID;
X    strcpy(nmp->n_name, spar1);
X#ifdef OUT_AZ
X    strcat(nmp->n_name, "#");
X#endif
X}
X
X/*
X * Transform hard ++,-- to equivalent trees
X * (for us, floats or fields)
X */
Xm_incdec(np)
XNODEP np;
X{
X    if (np->g_type != E_UNARY)
X         return 0;
X    if (np->g_ty != ET_F && !isfield(np->n_left))
X         return 0;
X
X    ipar2 = 0;
X    switch (np->g_token) {
X    case DOUBLE '+':
X         ipar1 = ASSIGN '+';
X         spar1 = "+=";
X         break;
X    case DOUBLE '-':
X         ipar1 = ASSIGN '-';
X         spar1 = "-=";
X         break;
X    case POSTINC:
X         ipar1 = DOUBLE '+';
X         spar1 = "++";
X         ipar2 = '-';
X         spar2 = "-";
X         break;
X    case POSTDEC:
X         ipar1 = DOUBLE '-';
```

```
X         spar1 = "--";
X         ipar2 = '+';
X         spar2 = "+";
X         break;
X    default:
X         return 0;
X    }
X    return 1;
X}
X
Xincdec(np)
Xregister NODEP np;
X{
X    NODEP t1;
X    NODEP onep;
X
X    onep = newgcon(1L, ET_S, SIZE_I);
X
X    if (ipar2 == 0) {      /* easy case, ++X becomes X+=1 */
X         np->g_token = ipar1;
X         np->g_type = E_BIN;
X         np->n_right = onep;
X         strcpy(np->n_name, spar1);
X         return;
X    }
X
X    /* hard case, X++ becomes (++X - 1) */
X    t1 = copyone(np);
X    t1->n_left = np->n_left;
X    np->n_left = t1;
X    np->n_right = onep;
X    np->g_type = E_BIN;
X    np->g_token = ipar2;
X    strcpy(np->n_name, spar2);
X
X    t1->g_token = ipar1;
X    strcpy(t1->n_name, spar1);
X}
X
X/*
X * Transform hard op= trees to equivalent '=' trees
X * (in this case, all floats, long or char *%/, fields)
X */
Xm_hardas(np)
XNODEP np;
X{
X    int op;
X
X    if (np->g_type != E_BIN)
X         return 0;
X    op = np->g_token;
X    if (isassign(op))
X         op -= ASSIGN 0;
```

```
X    else
X         return 0;
X    if (xstronger(np->n_right, np->n_left) &&
X         magicop(op) == 0)
X         return 1;
X    if (np->g_ty == ET_F || isfield(np->n_left))
X         return 1;
X    if (np->g_sz == 4 || np->g_sz == 1)
X         switch (op) {
X         case '*':
X         case '/':
X         case '%':
X              return 1;
X         }
X    return 0;
X}
X
Xhardas(np)
XNODEP np;
X{
X    NODEP opp, newl;
X    NODEP copynode();
X
X    if (m_vhard(np)) {
X         vhard(np);
X         return;
X    }
X
X    opp = copyone(np);
X    newl = copynode(np->n_left);
X    opp->n_right = np->n_right;
X    np->n_right = opp;
X    opp->n_left = newl;
X
X    np->g_token = '=';
X    strcpy(np->n_name, "unfold");
X
X    opp->g_token -= (ASSIGN 0);
X    bmaxty(opp);
X}
X
X/*
X * Check for lhs of op= that have side effects or are complex
X */
Xm_vhard(np)
XNODEP np;
X{
X    NODEP lp = np->n_left;
X
X    while (lp->g_token == '.')
X         lp = lp->n_left;
X    if (lp->g_token != STAR)
X         return 0;
```

```
X    return isvhard(lp->n_left);
X}
X
Xisvhard(np)
XNODEP np;
X{
X    NODEP rp;
X
Xdescend:
X    switch (np->g_type) {
X    case E_LEAF:
X         return 0;
X    case E_UNARY:
X         switch (np->g_token) {
X         case '(':
X         case DOUBLE '+':
X         case DOUBLE '-':
X         case POSTINC:
X         case POSTDEC:
X              return 1;
X         default:
X              np = np->n_left;
X              goto descend;
X         }
X    case E_BIN:
X         switch (np->g_token) {
X         case '+':
X         case '-':
X              rp = np->n_right;
X              if (rp->g_token == ICON && np->g_ty != ET_F) {
X                   np = np->n_left;
X                   goto descend;
X              }
X              /* fall through */
X         default:
X              return 1;
X         }
X    }
X}
X
Xvhard(np)
XNODEP np;
X{
X    NODEP starp;
X    NODEP atree, btree;
X    NODEP t1, t2;
X    register NODEP opp;
X    NODEP tmp_var();
X
X    starp = np->n_left;
X    while (starp->g_token == '.')
X         starp = starp->n_left;
X    atree = starp->n_left;
```

```
X    btree = np->n_right;
X    t1 = tmp_var(ET_U, SIZE_P);
X    t2 = copyone(t1);
X    starp->n_left = t2;
X
X    opp = copyone(t1);
X    opp->g_type = E_BIN;
X    opp->g_token = '=';
X    strcpy(opp->n_name, "=");
X    opp->n_right = atree;
X    opp->n_left = t1;
X
X    comma_r(np, opp);
X}
X
Xcomma_r(topp, lp)
XNODEP topp, lp;
X{
X    register NODEP newp;
X
X    newp = copyone(topp);
X    topp->g_token = ',';
X    strcpy(topp->n_name, ",");
X    newp->n_left = topp->n_left;
X    newp->n_right = topp->n_right;
X    topp->n_left = lp;
X    topp->n_right = newp;
X}
X
XNODEP
Xtmp_var(ty, sz)
X{
X    register NODEP t1;
X
X    t1 = allocnode();
X    t1->g_token = OREG;
X    t1->g_type = E_LEAF;
X    t1->g_rno = AREG+6;
X    t1->g_ty = ty;
X    t1->g_sz = sz;
X    t1->g_offs = - tmp_alloc(sz);
X    strcpy(t1->n_name, "tmp_v");
X    return t1;
X}
X
X/* X op= Y where Y's type is stronger than X's
X    either unfold it or (default)
X    cast Y to weaker type (+ or -)
X*/
X
Xmagicop(op)
X{
X    switch (op) {
```

```
X    case '+':
X    case '-':
X    case DOUBLE '<':
X    case DOUBLE '>':
X    case '&':
X    case '|':
X    case '^':
X         return 1;
X    }
X    return 0;
X}
X
Xstronger(xp, yp)
XNODEP xp, yp;
X{
X    if (xp->g_sz > yp->g_sz ||
X         (xp->g_sz == yp->g_sz && xp->g_ty > yp->g_ty))
X         return 1;
X    return 0;
X}
X
X/* stronger with ET_S and ET_U considered equal */
Xxstronger(xp, yp)
XNODEP xp, yp;
X{
X    if (xp->g_sz > yp->g_sz ||
X         (xp->g_ty == ET_F && yp->g_ty != ET_F))
X         return 1;
X    return 0;
X}
X
X/* give np the type of the stronger child */
Xbmaxty(np)
XNODEP np;
X{
X    NODEP lp = np->n_left, rp = np->n_right;
X
X    if (stronger(lp, rp))
X         rp = lp;
X    np->g_ty = rp->g_ty;
X    np->g_sz = gwiden(rp->g_sz);
X}
X
X/*
X *  Change floating compares to inline call
X */
Xm_fcmp(np)
XNODEP np;
X{
X         /* already made L and R same with casts */
X    if (np->g_type != E_BIN || np->n_left->g_ty != ET_F)
X         return 0;
X    switch (np->g_token) {
```

```
X    case '<':
X         spar2 = "lt";
X         return 1;
X    case '>':
X         spar2 = "gt";
X         return 1;
X    case DOUBLE '=':
X         spar2 = "eq";
X         return 1;
X    case NOTEQ:
X         spar2 = "ne";
X         return 1;
X    case GTEQ:
X         spar2 = "ge";
X         return 1;
X    case LTEQ:
X         spar2 = "le";
X         return 1;
X    }
X    return 0;
X}
X
Xfcmp(np)
Xregister NODEP np;
X{
X    register NODEP tp;
X
X    spar1 = "%fpcmp";
X    inline(np);
X
X    tp = copyone(np);
X    tp->n_left = np->n_left;
X    tp->n_right = np->n_right;
X    np->n_left = tp;
X
X    np->n_right = NULL;
X    np->g_type = E_UNARY;
X    np->g_token = CMPBR;
X    sprintf(np->n_name, spar2);
X}
X
X/*
X *  Remove useless binary operations with identity constant
X */
Xm_eident(np)
XNODEP np;
X{
X    NODEP rp = np->n_right;
X    long l;
X    int i, op;
X
X    if (np->g_type != E_BIN)
X         return 0;
```

```
X    if (np->g_ty == ET_F)
X         return 0;
X    while (rp->g_token == TCONV && rp->g_ty != ET_F)
X         rp = rp->n_left;
X    if (rp->g_token != ICON)
X         return 0;
X    l = rp->g_offs;
X    if (l < 0 || l > 1)
X         return 0;
X
X    op = np->g_token;
X    if (isassign(op))
X         op -= ASSIGN 0;
X    switch (op) {
X    case '+':
X    case '-':
X    case DOUBLE '<':
X    case DOUBLE '>':
X    case '|':
X    case '^':
X         i = 0;    break;
X    case '*':
X    case '/':
X         i = 1;  break;
X    default:
X         return 0;
X    }
X    if (l != i)
X         return 0;
X    return 1;
X}
X
Xeident(np)
XNODEP np;
X{
X    NODEP lp = np->n_left, rp = np->n_right;
X
X    freenode(rp);
X
X    lcpy(np, lp, sizeof(NODE)/4);
X
X    freeunit(lp);
X}
X
X#define MAXLOOK   8
X
X/*
X * Change certain mult or div to equivalent shift
X */
Xm_md_shf(np)
XNODEP np;
X{
X    NODEP rp = np->n_right;
```

```
X    long l;
X    register i, j;
X
X    if (np->g_type != E_BIN)
X         return 0;
X    if (np->g_ty == ET_F)
X         return 0;
X    while (rp->g_token == TCONV && rp->g_ty != ET_F)
X         rp = rp->n_left;
X    if (rp->g_token != ICON)
X         return 0;
X
X    switch (np->g_token) {
X    case '*':
X         ipar1 = DOUBLE '<';  break;
X    case '/':
X         ipar1 = DOUBLE '>';  break;
X    case ASSIGN '*':
X         ipar1 = ASSIGN DOUBLE '<';  break;
X    case ASSIGN '/':
X         ipar1 = ASSIGN DOUBLE '>';  break;
X    default:
X         return 0;
X    }
X
X    l = rp->g_offs;
X    if (l < 2 || l > (1<<MAXLOOK))
X         return 0;
X    i = l;
X    for (j=1; j<=MAXLOOK; j++)
X         if (i == 1<<j) {
X              ipar2 = j;
X              return 1;
X         }
X    return 0;
X}
X
Xmd_shf(np)
XNODEP np;
X{
X    NODEP rp = np->n_right;
X
X    np->g_token = ipar1;
X    while (rp->g_token == TCONV)
X         rp = rp->n_left;
X    rp->g_offs = ipar2;
X}
X
Xm_fldas(np)
XNODEP np;
X{
X    if (np->g_type != E_BIN)
X         return 0;
```

```
X    if (np->g_token == '=' && isfield(np->n_left))
X         return 1;
X    return 0;
X}
X
Xfldas(np)
Xregister NODEP np;
X{
X    NODEP lp = np->n_left;
X
X    np->g_fldw = lp->g_fldw;
X    np->g_fldo = lp->g_fldo;
X    np->g_token = FIELDAS;
X
X    lp->g_fldw = 0;
X}
X
Xred_con(np)
Xregister NODEP np;
X{
X    while (np->g_token == TCONV)
X         np = np->n_left;
X    if (np->g_token == ICON || np->g_token == FCON)
X         return 1;
X    return 0;
X}
END_OF_FILE
if test 15002 -ne `wc -c <'hcc/GUNK.C'`; then
    echo shar: \"'hcc/GUNK.C'\" unpacked with wrong size!
fi
# end of 'hcc/GUNK.C'
fi
if test -f 'hcc/P2.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/P2.C'\"
else
echo shar: Extracting \"'hcc/P2.C'\" \(14397 characters\)
sed "s/^X//" >'hcc/P2.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  p2.c
X *
X *  Expression tree routines.
X *
X *  Constant folding, typing of nodes, simple transformations.
X */
```

```
X
X#include <stdio.h>
X#include "param.h"
X#include "tok.h"
X#include "nodes.h"
X#include "cookie.h"
X
X#if MMCC
Xoverlay "pass2"
X#endif
X
Xextern int xflags[];
X#define debug xflags['t'-'a']
X
Xextern nmerrors;
XNODEP bas_type();
X
Xdo_expr(np, cookie)
XNODE *np;
X{
X    if (np == NULL)
X         return;
X/*  include if want only one error at a time
X    if (nmerrors) {
X         freenode(np);
X         return;
X    }
X*/
X    p2_expr(&np);
X    genx(np, cookie);
X}
X
Xp2_expr(npp)
XNODEP *npp;
X{
X    NODEP np = *npp;
X
X    if (np == NULL) return;
X    if (debug > 1) {
X         printf("P2 enter");
X         printnode(np);
X    }
X    confold(npp,0);
X    np = *npp;
X    form_types(np);
X    if (debug) {
X         printf("p2_expr");
X         printnode(np);
X    }
X    return;
X}
X
Xform_types(np)
```

```
XNODEP np;
X{
X
X    if (np == NULL) return;
X    switch (np->e_type) {
X    case E_SPEC:
X         switch (np->e_token) { /* special cases */
X         case '.':
X         case ARROW:
X              form_types(np->n_left);
X              sel_type(np);
X              return;
X         case '(':
X              if (np->n_right) {
X                   form_types(np->n_right);    /* args */
X                   np->e_type = E_BIN;
X              } else
X                   np->e_type = E_UNARY;
X              fun_type(np);
X              return;
X         }
X         /* fall through */
X    case E_BIN:
X         form_types(np->n_left);
X         form_types(np->n_right);
X         b_types(np);
X         break;
X
X    case E_UNARY:
X         form_types(np->n_left);
X         u_types(np);
X         break;
X
X    case E_LEAF:
X         l_types(np);
X         break;
X    }
X}
X
X/* (fun) (args) */
Xfun_type(np)
XNODEP np;
X{
X    NODEP lp, typ;
X    NODEP allsyms(), new_fun();
X
X    lp = np->n_left;
X    if (lp->e_token == ID) { /* may be new ID */
X         typ = allsyms(lp);
X         if (typ == NULL)
X              typ = new_fun(lp);
X         typ = typ->n_tptr;
X         lp->n_tptr = typ;
```

```
X         lp->n_flags |= N_COPYT;
X    } else {
X         form_types(lp);
X         typ = lp->n_tptr;
X    }
X    if (typ->t_token != '(') {  /* fun ret ? */
X         error("call non-fun");
X         goto bad;
X    }
X    typ = typ->n_tptr;
X    goto good;
Xbad:
X    typ = bas_type(K_INT);
Xgood:
X    np->n_tptr = typ;
X    np->n_flags |= N_COPYT;
X}
X
X/* (struct|union) (. or ->) ID */
Xsel_type(xp)
XNODEP xp;
X{
X    NODEP np, sup;
X    int tok;
X    NODEP rv;
X    NODEP llook();
X
X    np = xp->n_right;
X    sup = xp->n_left->n_tptr;
X    tok = xp->e_token;
X
X/* already checked that np->e_token == ID */
X    if (tok == ARROW) {
X         if (sup->t_token != STAR) {
X              error("(non pointer)->");
X              goto bad;
X         }
X         sup = sup->n_tptr;
X    }
X    if (sup->t_token != K_STRUCT && sup->t_token != K_UNION) {
X         error("select non-struct");
X         goto bad;
X    }
X    rv = llook(sup->n_right, np);
X    if (rv == NULL) {
X         error("? member ID");
X         goto bad;
X    }
X    xp->e_offs = rv->e_offs;
X    if (rv->e_fldw) {
X         xp->e_fldw = rv->e_fldw;
X         xp->e_fldo = rv->e_fldo;
X    }
```

```
X    rv = rv->n_tptr;
X    goto good;
Xbad:
X    rv = bas_type(K_INT);
Xgood:
X    xp->n_tptr = rv;
X    xp->n_flags |= N_COPYT;
X
X    /* change to UNARY op */
X    xp->e_type = E_UNARY;
X    freenode(np);
X    xp->n_right = NULL;
X
X    /* change ARY OF to PTR TO */
X    if (rv->t_token == '[')
X         see_array(xp);
X}
X
Xl_types(np)
Xregister NODE *np;
X{
X    NODEP allsyms();
X    register NODE *tp;
X
X    switch (np->e_token) {
X    case ID:  /* already did see_id */
X         if (np->n_tptr->t_token == '[')  /* change to &ID */
X              see_array(np);
X         return;
X    case ICON:
X         tp = bas_type(icon_ty(np));
X         break;
X    case FCON:
X         tp = bas_type(K_DOUBLE);
X         break;
X    case SCON:
X         tp = bas_type(SCON);
X         break;
X    default:
X         errors("Weird leaf",np->n_name);
X    bad:
X         tp = bas_type(K_INT);
X    }
X    np->n_tptr = tp;
X    np->n_flags |= N_COPYT;
X}
X
Xu_types(np)
XNODEP np;
X{
X    NODEP tp;
X    NODEP lp = np->n_left;
X    NODEP normalty();
```

```
X
X    tp = lp->n_tptr;   /* default */
X
X    switch (np->e_token) {
X    case DOUBLE '+':
X    case DOUBLE '-':
X    case POSTINC:
X    case POSTDEC:
X         mustlval(lp);
X         mustty(lp, R_SCALAR);
X         if (tp->t_token == STAR)
X              np->e_offs = tp->n_tptr->t_size;
X         else
X              np->e_offs = 1;
X         break;
X    case STAR:
X         if (mustty(lp, R_POINTER)) goto bad;
X         tp = tp->n_tptr;
X         np->n_tptr = tp;
X         np->n_flags |= N_COPYT;
X
X         /* Ary of to Ptr to */
X         if (tp->t_token == '[')
X              see_array(np);
X         return;
X    case UNARY '&':
X         mustlval(lp);
X         tp = allocnode();
X         tp->n_tptr = lp->n_tptr;
X         tp->n_flags |= N_COPYT;
X         tp->t_token = STAR;
X         sprintf(tp->n_name, "Ptr to");
X         tp->t_size = SIZE_P;
X         np->n_tptr = tp;
X         return;       /* no COPYT */
X    case UNARY '-':
X         mustty(lp, R_ARITH);
X         tp = normalty(lp, NULL);
X         break;
X    case TCONV:
X         mustty(lp, R_SCALAR);
X         if (np->n_tptr->t_token != K_VOID)
X              mustty(np, R_SCALAR);
X         return;       /* type already specified */
X    case '!':
X         mustty(lp, R_SCALAR);
X         tp = bas_type(K_INT);
X         break;
X    case '~':
X         mustty(lp, R_INTEGRAL);
X         tp = normalty(lp, NULL);
X         break;
X    default:
```

```
X         error("bad unary type");
X    bad:
X         tp = bas_type(K_INT);
X    }
X    np->n_tptr = tp;
X    np->n_flags |= N_COPYT;
X}
X
Xb_types(np)
XNODEP np;
X{
X    NODEP tp;
X    NODEP lp, rp;
X    NODEP normalty(), addty(), colonty();
X    int op;
X
X    op = np->e_token;
X    if (isassign(op)) {
X         mustlval(np->n_left);
X         op -= (ASSIGN 0);
X    }
X
X    lp = np->n_left;
X    rp = np->n_right;
X    tp = bas_type(K_INT);
X    switch (op) {
X    case '*':
X    case '/':
X         mustty(lp, R_ARITH);
X         mustty(rp, R_ARITH);
X         tp = normalty(lp,rp);
X         break;
X    case '%':
X    case '&':
X    case '|':
X    case '^':
X         mustty(lp, R_INTEGRAL);
X         mustty(rp, R_INTEGRAL);
X         tp = normalty(lp,rp);
X         break;
X    case '+':
X    case '-':
X         mustty(lp, R_SCALAR);
X         mustty(rp, R_SCALAR);
X         tp = addty(np);
X         break;
X    case DOUBLE '<':
X    case DOUBLE '>':
X         mustty(lp, R_INTEGRAL);
X         mustty(rp, R_INTEGRAL);
X         tp = normalty(lp, NULL);
X         break;
X    case '<':
```

```
X    case '>':
X    case LTEQ:
X    case GTEQ:
X    case DOUBLE '=':
X    case NOTEQ:
X         mustty(lp, R_SCALAR);
X         mustty(rp, R_SCALAR);
X         chkcmp(np);
X         break;         /* INT */
X    case DOUBLE '&':
X    case DOUBLE '|':
X         mustty(lp, R_SCALAR);
X         mustty(rp, R_SCALAR);
X         break;         /* INT */
X    case '?':
X         mustty(lp, R_SCALAR);
X         tp = rp->n_tptr;
X         break;
X    case ':':
X         if (same_type(lp->n_tptr, rp->n_tptr)) {
X              tp = lp->n_tptr;
X              break;
X         }
X         mustty(lp, R_SCALAR);
X         mustty(rp, R_SCALAR);
X         tp = colonty(np);
X         break;
X    case '=':
X         mustlval(lp);
X         mustty(lp, R_ASSN);
X         asn_chk(lp->n_tptr, rp);
X         tp = lp->n_tptr;
X         break;
X    case ',':
X         tp = rp->n_tptr;
X         break;
X    default:
X         error("bad binary type");
X    bad:
X         tp = bas_type(K_INT);
X    }
X    if (isassign(np->e_token)) {
X         /* ignore normal result -- result is left type */
X         tp = lp->n_tptr;
X    }
X    np->n_tptr = tp;
X    np->n_flags |= N_COPYT;
X}
X
Xlong
Xconlval(np)
XNODEP np;
X{
```

```
X    long i;
X
X    confold(&np,0);
X    if (np->e_token == ICON) {
X         i = np->e_ival;
X         freenode(np);
X         return i;
X    }
X    error("need const expr");
X    return 0;
X}
X
Xconxval(np)
XNODEP np;
X{
X    return (int)conlval(np);
X}
X
Xconfold(npp,spec)
XNODEP *npp;
X{
X    NODEP np;
X    NODEP tp, onp;
X    int tok,spl,spr;
X    long l;
X
X    np = *npp;
X    if (np == NULL) return;
X    switch (np->e_type) {
X    case E_LEAF:
X              lcanon(np,spec);
X              return;
X    case E_UNARY:
X              confold(&np->n_left,0);
X              ucanon(np);
X              return;
X    case E_BIN:
X              confold(&np->n_left,0);
X    /* delay confold on the right tree */
X              switch (np->e_token) {
X              case DOUBLE '|':
X                   l = np->n_left->e_ival;
X                   tp = np;
X                   goto l_or_r;
X              case DOUBLE '&':
X                   l = ! np->n_left->e_ival;
X                   tp = np;
X                   goto l_or_r;
X              case '?':
X                   l = np->n_left->e_ival;
X                   tp = np->n_right;  /* ':' node */
X         l_or_r:
X                   tok = np->n_left->e_token;
```

```
X                  if (tok != ICON) {
X                       confold(&np->n_right,0);
X                       return;
X                  }
X                  onp = np;
X                  if (l) {  /* take true side */
X                       np = tp->n_left;
X                       tp->n_left = NULL;
X                  } else {  /* take false side */
X                       np = tp->n_right;
X                       tp->n_right = NULL;
X                  }
X                  freenode(onp);
X                  confold(&np,0);
X                  *npp = np;
X                  return;
X             }
X             confold(&np->n_right,0);
X             bcanon(np);
X             if (np->e_flags & C_AND_A)
X                  b_assoc(np);
X             return;
X    case E_SPEC:
X         tok = np->e_token;
X         spl = spr = 0;
X         switch (tok) {
X         case '(':
X              spl = tok;    /* new name allowed */
X              break;
X         case '.':
X         case ARROW:
X              spr = tok;    /* look in struct sym.tab. */
X              break;
X         }
X         confold(&np->n_left,spl);
X         confold(&np->n_right,spr);
X         return;
X    }
X}
X
Xnewicon(np,x,nf)
XNODE *np;
Xlong x;
X{
X    np->e_token = ICON;
X    np->e_ival = x;
X    np->e_flags = nf;
X    sprintf(np->n_name, "%ld", x);
X    np->e_type = E_LEAF;
X    if (np->n_left) {
X         freenode(np->n_left);
X         np->n_left = NULL;
X    }
```

```
X    if (np->n_right) {
X         freenode(np->n_right);
X         np->n_right = NULL;
X    }
X}
X
Xnewfcon(np,x,nf)
XNODE *np;
Xdouble x;
X{
X    np->e_token = FCON;
X    np->e_fval = x;
X    np->e_flags = nf;
X    sprintf(np->n_name, FLTFORM, x);
X    np->e_type = E_LEAF;
X    if (np->n_left) {
X         freenode(np->n_left);
X         np->n_left = NULL;
X    }
X    if (np->n_right) {
X         freenode(np->n_right);
X         np->n_right = NULL;
X    }
X}
X
X/* LEAF */
X/* sptok is token if E_SPEC node is parent
X   and dont want to look at ID yet */
Xlcanon(np,sptok)
XNODE *np;
X{
X    NODE *tp;
X    NODEP allsyms();
X    long x;
X
X    if (np->e_token == ID) {
X         if (sptok)
X              return;
X         see_id(np);
X         return;
X    }
X    if (np->e_token == TSIZEOF) {
X         tp = np->n_tptr;
X         x = tp->t_size;
X         np->n_tptr = NULL;
X         if ((np->n_flags & N_COPYT) == 0)
X              freenode(tp);
X         newicon(np, x, 0);
X    }
X}
X
X/* UNARY */
Xucanon(np)
```

```
XNODE *np;
X{
X    NODE *tp;
X    long x,l;
X    int lflags = 0;
X
X    if (np->e_token == K_SIZEOF) {
X         tp = np->n_left;
X         confold(&tp,0);
X         form_types(tp);
X         tp = tp->n_tptr;
X         x = tp->t_size;
X         goto out;
X    }
X
X    if (np->n_left->e_token == FCON) {
X         if (np->e_token == UNARY '-')
X              newfcon(np, -(np->n_left->e_fval));
X         return;
X    }
X    if (np->n_left->e_token != ICON)
X         return;
X    l = np->n_left->e_ival;
X    lflags = np->n_left->e_flags;
X    switch (np->e_token) {
X    case UNARY '-':
X              x = -l;       break;
X    case '~':
X              x = ~l;       break;
X    case '!':
X              x = !l;       break;
X    default:
X         return;
X    }
Xout:
X    newicon(np, x, lflags);
X}
X
Xbcanon(np)
Xregister NODE *np;
X{
X    int ltok, rtok;
X    double l,r;
X    NODEP tp;
X
X    ltok = np->n_left->e_token;
X    rtok = np->n_right->e_token;
X    if (ltok != ICON && ltok != FCON)
X         return;
X    if (rtok != ICON && rtok != FCON) {
X    /* left is ?CON, right is not */
X         if (np->e_flags & (C_AND_A|C_NOT_A)) {
X         /* reverse sides  - put CON on right */
```

```
X             tp = np->n_left;
X             np->n_left = np->n_right;
X             np->n_right = tp;
X             if (np->e_flags & C_NOT_A)
X                  swt_op(np);
X         }
X         return;
X    }
X    if (ltok == ICON && rtok == ICON) {
X         b2i(np);
X         return;
X    }
X    if (ltok == FCON)
X         l = np->n_left->e_fval;
X    else
X         l = (double)np->n_left->e_ival;
X    if (rtok == FCON)
X         r = np->n_right->e_fval;
X    else
X         r = (double)np->n_right->e_ival;
X    b2f(np,l,r);
X}
X
X/* canon for assoc. & comm. op */
X/* this code will almost never be executed, but it was fun. */
Xb_assoc(np)
XNODEP np;
X{
X    NODEP lp, rp;
X    int tok;
X
X    lp = np->n_left;
X    if (lp->e_token != np->e_token)
X         return;
X    /* left is same op as np */
X    rp = np->n_right;
X    tok = lp->n_right->e_token;
X    if (tok != ICON && tok != FCON)
X         return;
X    /* left.right is ?CON */
X    tok = rp->e_token;
X    if (tok == ICON || tok == FCON) {
X         /* have 2 CONS l.r and r -- put together on r */
X         NODEP    ep;
X         ep = lp->n_left;
X         np->n_left = ep;
X         np->n_right = lp;
X         lp->n_left = rp;
X         /* can now fold 2 CONS */
X         bcanon(lp);
X    } else {
X         /* have 1 CON at l.r -- move to top right */
X         NODEP    kp;
```

```
X         kp = lp->n_right;
X         lp->n_right = rp;
X         np->n_right = kp;
X    }
X}
X
X/* switch pseudo-commutative op */
Xswt_op(np)
XNODEP np;
X{
X    int newtok;
X    char *newnm;
X
X    switch (np->e_token) {
X    case '<': newtok = '>'; newnm = ">";  break;
X    case '>': newtok = '<'; newnm = "<";  break;
X    case LTEQ:    newtok = GTEQ;     newnm = ">="; break;
X    case GTEQ:    newtok = LTEQ;     newnm = "<="; break;
X    default:
X         return;
X    }
X    np->e_token = newtok;
X    strcpy(np->n_name, newnm);
X}
X
X/* BINARY 2 ICON's */
Xb2i(np)
Xregister NODE *np;
X{
X    register long l,r,x;
X    int newflags,lflags;
X
X    newflags = 0;
X
X    r = np->n_right->e_ival;
X    newflags = np->n_right->e_flags;
X
X    l = np->n_left->e_ival;
X    lflags = np->n_left->e_flags;
X    newflags = newflags>lflags ? newflags : lflags;
X
X    switch (np->e_token) {
X    case '+':
X              x = l+r;  break;
X    case '-':
X              x = l-r;  break;
X    case '*':
X              x = l*r;  break;
X    case '/':
X              x = l/r;  break;
X    case '%':
X              x = l%r;  break;
X    case '>':
```

```
X             x = l>r;  break;
X    case '<':
X             x = l<r;  break;
X    case LTEQ:
X             x = l<=r; break;
X    case GTEQ:
X             x = l>=r; break;
X    case DOUBLE '=':
X             x = l==r; break;
X    case NOTEQ:
X             x = l!=r; break;
X    case '&':
X             x = l&r;  break;
X    case '|':
X             x = l|r;  break;
X    case '^':
X             x = l^r;  break;
X    case DOUBLE '<':
X             x = l<<r; break;
X    case DOUBLE '>':
X             x = l>>r; break;
X    default:
X         return;
X    }
X    newicon(np, x, newflags);
X}
X
X/* BINARY 2 FCON's */
Xb2f(np,l,r)
Xregister NODE *np;
Xdouble l,r;
X{
X    register double x;
X    int ix, isint;
X
X    isint = 0;
X
X    switch (np->e_token) {
X    case '+':
X             x = l+r;  break;
X    case '-':
X             x = l-r;  break;
X    case '*':
X             x = l*r;  break;
X    case '/':
X             x = l/r;  break;
X    case '>':
X             ix = l>r; isint++;  break;
X    case '<':
X             ix = l<r; isint++;  break;
X    case LTEQ:
X             ix = l>=r;    isint++;  break;
X    case GTEQ:
```

```
X             ix = l<=r;    isint++; break;
X    case DOUBLE '=':
X             ix = l==r;    isint++; break;
X    case NOTEQ:
X             ix = l!=r;    isint++; break;
X    default:
X         return;
X    }
X    if (isint)
X         newicon(np, (long)ix, 0);
X    else
X         newfcon(np, x);
X}
X
Xsame_type(a,b)
Xregister NODE *a, *b;
X{
Xmore:
X    if (a == b)
X         return 1;
X    if (a == NULL || b == NULL)
X         return 0;
X    if (a->t_token != b->t_token)
X         return 0;
X    if (a->t_token != STAR && a->t_size != b->t_size)
X         return 0;
X    a = a->n_tptr;
X    b = b->n_tptr;
X    goto more;
X}
X
Xsee_id(np)
XNODEP np;
X{
X    NODEP tp;
X    NODEP allsyms(), def_type();
X
X    tp = allsyms(np);
X    if (tp == NULL) {
X         errorn("undefined:", np);
X         tp = def_type();
X         goto out;
X    }
X    switch (tp->e_sc) {
X    case ENUM_SC:
X         newicon(np, tp->e_ival, 0);
X         return;
X    case K_REGISTER:
X         np->e_rno = tp->e_rno;
X         /* fall through */
X    default:
X         np->e_sc = tp->e_sc;
X         np->e_offs = tp->e_offs;
```

```
X         tp = tp->n_tptr;
X    }
Xout:
X    np->n_tptr = tp;
X    np->n_flags |= N_COPYT;
X
X    /* special conversions */
X    if (tp->t_token == '(')
X         insptrto(np);
X}
X
Xinsptrto(np)
XNODEP np;
X{
X    NODEP op, copyone();
X
X    op = copyone(np);
X
X    np->n_left = op;
X    np->e_token = UNARY '&';
X    np->e_type = E_UNARY;
X    strcpy(np->n_name, "&fun");
X    np->n_flags &= ~N_COPYT;
X}
X
X/* np points to ID or STAR or '.' node
X    tptr is a COPY
X    tptr token is '[' */
X
Xsee_array(np)
XNODEP np;
X{
X    NODEP tp, copyone();
X
X    tp = copyone(np);
X    tp->n_left = np->n_left;
X    tp->n_tptr = tp->n_tptr->n_tptr;
X
X    np->n_left = tp;
X    np->e_token = UNARY '&';
X    np->e_type = E_UNARY;
X    strcpy(np->n_name, "&ary");
X    arytoptr(np);
X/* leave old size
X    np->n_tptr->t_size = SIZE_P;
X*/
X}
END_OF_FILE
if test 14397 -ne `wc -c <'hcc/P2.C'`; then
    echo shar: \"'hcc/P2.C'\" unpacked with wrong size!
fi
# end of 'hcc/P2.C'
fi
```

```
if test -f 'hcc/PRE.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'hcc/PRE.C'\"
else
echo shar: Extracting \"'hcc/PRE.C'\" \(14315 characters\)
sed "s/^X//" >'hcc/PRE.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Johann Ruegg
X * 
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X *
X *  pre.c
X *
X *  preprocessor for the compiler
X *
X *  Handles all preprocessor (#) commands and
X *    looks up keywords
X *
X *  Interface:
X *       getnode() returns next "token node"
X */
X
X#include <stdio.h>
X#include "param.h"
X#include "tok.h"
X#include "nodes.h"
X
X#if CC68
XFILE *fopenb();
X#define fopen fopenb
X#endif
X
Xextern struct tok curtok;
Xextern char curstr[];
X
X#define TK_SEENL  1    /* want to see NL token */
X#define TK_SEEWS  2    /* want to see WS token */
X#define TK_ONLY1  4    /* only want 1st token on line */
X#define TK_LTSTR  8    /* '<' starts a string */
X#define TK_NOESC  16   /* dont do '\' escapes in string */
X
Xextern int tk_flags, sawnl;
X
X#ifndef ACK_HOST
XNODE *deflist[NHASH];
XNODE *holdtok;
Xint iflevel, iftruth, ifnest, in_if_x, skip_id;
X#else
X#if PART_1
```

```
XNODE *deflist[NHASH];
XNODE *holdtok;
Xextern int iflevel, iftruth, ifnest, in_if_x, skip_id;
X#else
Xextern NODE *deflist[NHASH];
Xextern NODE *holdtok;
Xint iflevel, iftruth, ifnest, in_if_x, skip_id;
X#endif
X#endif
X
Xextern lineno;
Xextern char *inname;
Xextern FILE *input;
X
XNODE *hlook(), *llook();
XNODEP tok_to_node();
XNODE *copylist();
X
X#ifdef DEBUG
Xextern int oflags[];
X#define debugd    oflags['d'-'a']
X#define debugt    oflags['t'-'a']
X#endif
X
X#ifndef ACK_HOST
X#define PART_1    1
X#define PART_2    1
X#endif
X
X#if PART_1
X
XNODEP
Xhi_node()
X{
X    register NODEP rv;
X
X    /* node from hold queue ? */
X    if (holdtok) {
X#ifdef DEBUG
X        if (debugd > 2) {
X             printf("Holdqueue");
X             printnode(holdtok);
X        }
X#endif
X        rv = holdtok;
X        holdtok = rv->n_next;
X        rv->n_next = NULL;
X        return rv;
X    }
X    /* node from input */
Xagain:
X    while (iflevel && !iftruth)
X        skiplines();
```

```
X    if (nxttok()==0)
X         return NULL;
X    if (curtok.tnum == '#') {
X         dopound(0);
X         goto again;
X    }
X    rv = tok_to_node();
X    return rv;
X}
X
XNODEP
Xgetnode()
X{
X    register NODEP rv;
X    NODEP dp;
X
Xagain:
X    rv = hi_node();
X    if (rv == NULL) {
X         rv = allocnode();
X         rv->e_token = EOFTOK;
X         strcpy(rv->n_name, "*EOF*");
X    } else
X    if (rv->e_token == ID) {
X         if (in_if_x && strcmp(rv->n_name, "defined") == 0) {
X              skip_id = 1;
X              goto out;
X         }
X         if (skip_id) {
X              skip_id = 0;
X              goto out;
X         }
X         if ((dp = hlook(deflist, rv)) != NULL) {
X              expand(dp);
X              freenode(rv);
X              goto again;
X         } else if (rv->n_name[0] == '_' && builtin(rv))
X              return rv;
X         else
X              kw_tok(rv);
X    }
Xout:
X#ifdef DEBUG
X    if (debugt) {
X         putchar('[');
X         put_nnm(rv);
X         printf("] ");
X    }
X#endif
X    return rv;
X}
X
Xbuiltin(np)
```

```
Xregister NODEP np;
X{
X    int rv;
X
X    if (strcmp(np->n_name, "__LINE__") == 0) {
X         np->e_token = ICON;
X         np->e_ival = lineno;
X         return 1;
X    }
X    else if (strcmp(np->n_name, "__FILE__") == 0) {
X         np->e_token = SCON;
X         nscpy(np, inname);
X         return 1;
X    }
X    return 0;
X}
X
Xskiplines()
X{
X    for (;;) {
X         if (nxttok()== 0)
X              return;
X         if (curtok.tnum == '#') {
X              dopound(1);
X              return;
X         }
X         tk_flags |= TK_ONLY1;
X    }
X}
X
Xstatic defnargs;
X
Xp_def()
X{
X    NODE *args;
X    NODE *val;
X    NODE *def;
X    NODE *def_rgs(), *def_val();
X
X    defnargs = -1;
X    args = NULL;
X    val = NULL;
X    nxttok();
X    if (curtok.tnum != ID) {
X         error("bad #define");
X         goto flush;
X    }
X    def = tok_to_node();
X
X    tk_flags |= TK_SEEWS;
X    nxttok();
X    switch (curtok.tnum) {
X    case '(':
```

```
X         defnargs = 0;
X         args = def_rgs();
X    case WS:
X         goto getval;
X    case NL:
X         goto dodef;
X    default:
X         error("bad #define");
X         goto flush;
X    }
Xgetval:
X    val = def_val();
Xdodef:
X    def->e_ival = defnargs;
X    define(def, val, args);
Xflush:
X    ;
X}
X
Xoptdef(s,as)
Xchar *s, *as;
X{
X    NODEP val;
X    NODEP def;
X    NODEP id_tok(), def_val();
X
X    defnargs = -1;
X    val = NULL;
X
X    def = id_tok(s);
X    chr_push(as);
X
X    tk_flags |= TK_SEENL;
X
X    val = def_val();
X
X    tk_flags = 0;
X
X    def->e_ival = defnargs;
X    define(def, val, NULL);
X}
X
X/*
Xoptundef(s)
Xchar *s;
X{
X    NODEP np, tp, id_tok();
X
X    np = id_tok(s);
X    tp = hlook(deflist, np);
X    if (tp != NULL)
X         tp->n_name[0] = '#';
X    freenode(np);
```

```
X}
X*/
X
Xsamedef(p1, p2)
XNODEP p1, p2;
X{
X    if (p1->e_ival != p2->e_ival)
X         return 0;
X    return same_list(p1->n_right, p2->n_right);
X}
X
Xsame_list(p1, p2)
XNODEP p1, p2;
X{
X    if (p1 == NULL)
X         return p2 == NULL;
X    if (p2 == NULL)
X         return 0;
X    if (l_cmp(p1, p2, sizeof(*p1)/sizeof(long)) != 0)
X         return 0;
X    return same_list(p1->n_left, p2->n_left);
X}
X
Xl_cmp(p1, p2, n)
XNODE *p1, *p2;
X{
X    if (xstrcmp(p1,p2) != 0)
X         return 1;
X    if (p1->e_token != p2->e_token ||
X         p1->e_ival != p2->e_ival)
X         return 1;
X    return 0;
X}
X
Xdefine(def, val, args)
XNODEP def, val, args;
X{
X    NODEP oldp;
X
X    if (args != NULL) {
X         argsmod(val, args);
X         freenode(args);
X    }
X    def->n_right = val;
X    if ((oldp = hlook(deflist, def)) != NULL) {
X         if (!samedef(oldp, def))
X              warnn("redefined", def);
X    }
X#ifdef DEBUG
X    if (debugd) {
X         printf("define (%d args)", (int)def->e_ival);
X         printnode(def);
X    }
```

```
X#endif
X    puthlist(deflist, def);
X}
X
Xargsmod(toks, args)
XNODEP toks, args;
X{
X    register NODE *np, *vp;
X
X    for (np=toks; np != NULL; np = np->n_next)
X         if (np->e_token == ID) {
X              vp = llook(args,np);
X              if (vp != NULL) {
X                   np->e_token = DPARAM;
X                   np->e_ival = vp->e_ival;
X                   sprintf(np->n_name, "\\%d", (int)np->e_ival);
X              }
X         }
X}
X
XNODE *
Xdef_rgs()
X{
X    NODE *rv;
X    NODE *tail;
X    NODE *np;
X
X    rv = NULL;
X    tail = NULL;
X    nxttok();
X    if (curtok.tnum == ')') {
X         goto out;
X    }
Xmore:
X    if (curtok.tnum != ID) {
X         error("expect ID");
X         goto bad;
X    }
X    np = tok_to_node();
X    np->e_ival = defnargs;  /* hold sequence number */
X    defnargs++;
X    if (tail == NULL) {     /* first one */
X         rv = np;
X         tail = np;
X    } else {           /* more */
X         tail->n_next = np;
X         tail = np;
X    }
X    nxttok();
X    if (curtok.tnum == ',') {
X         nxttok();
X         goto more;
X    }
```

```
X    if (curtok.tnum == ')')
X         goto out;
X    error("define arg syntax");
Xbad:
X    freenode(rv);
X    rv = NULL;
X    defnargs = 0;
Xout:
X    return rv;
X}
X
XNODE *
Xdef_val()
X{
X    NODE *rv;
X    NODE *tail;
X    NODE *np;
X
X    rv = NULL;
X    tail = NULL;
Xmore:
X    nxttok();
X    if (curtok.tnum == NL) {
X         goto out;
X/*
X    } else if (curtok.tnum == '\\') {
X         nxttok();
X         if (curtok.tnum != NL)
X              goto bad;
X         goto more;
X*/
X    }
X    np = tok_to_node();
X    if (tail == NULL) {    /* first one */
X         rv = np;
X         tail = np;
X    } else {           /* more */
X         tail->n_next = np;
X         tail = np;
X    }
X    goto more;
X
Xbad:
X    freenode(rv);
X    rv = NULL;
Xout:
X    return rv;
X}
X
XNODE *
Xgath1(sep)
Xint *sep;
X{
```

```
X    NODE *np, *rv, *tail;
X    int inparen;
X
X    inparen = 0;
X    rv = NULL;
X    tail = NULL;
Xmore:
X    np = hi_node();
X    if (np == NULL) {
X         goto bad;
X    }
X    switch (np->e_token) {
X    case ')':
X    case ',':
X         if (inparen) {     /* dont end, part of subexpr */
X              if (np->e_token == ')')
X                   inparen--;
X              break;
X         }
X         *sep = np->e_token;
X         freenode(np);
X         goto out;
X    case '(':
X         inparen++;
X         break;
X    }
X    if (tail == NULL) {     /* first one */
X         rv = np;
X         tail = np;
X    } else {           /* more */
X         tail->n_next = np;
X         tail = np;
X    }
X    goto more;
Xbad:
X    freenode(rv);
X    rv = NULL;
X    *sep = 0;
Xout:
X    return rv;
X}
X
XNODE *
Xgath_args(n)
X{
X    NODE *rv;
X    NODE *tail;
X    NODE *np;
X    int sep;
X    int getn;
X
X    getn = 0;
X    rv = NULL;
```

```
X    tail = NULL;
X    np = hi_node();
X    if (np->e_token != '(') {
X         error("expect (");
X         goto bad;
X    }
X    freenode(np);
X    if (n == 0) {
X         np = hi_node();
X         if (np->e_token != ')') {
X              error("expect )");
X              goto bad;
X         }
X         freenode(np);
X         return NULL;
X    }
Xmore:
X    np = gath1(&sep);
X    if (np == NULL) {
X         error("expect arg");
X         goto bad;
X    }
X    getn++;
X    if (tail == NULL) {    /* first one */
X         rv = np;
X         tail = np;
X    } else {          /* more */
X         tail->n_right = np;
X         tail = np;
X    }
X    if (sep) switch (sep) {
X    case ',':
X         goto more;
X    case ')':
X         if (getn != n) {
X              error("arg num mismatch");
X              goto bad;
X         }
X         goto out;
X    }
X    error("expand arg syntax");
Xbad:
X    freenode(rv);
X    rv = NULL;
Xout:
X    return rv;
X}
X
XNODE *
Xargfix(val, args, rt)
XNODE *val, *args;
XNODE **rt;
X{
```

```
X    register NODE *scan, *sub;
X    NODE *head;
X    NODE *tail, *back;
X    NODE *rthnode();
X    NODE *copylist();
X
X    head = val;
X    back = NULL;
X    for (scan = val; scan != NULL; back=scan, scan=scan->n_next)
X         if (scan->e_token == DPARAM) {
X              sub = rthnode(args, (int)scan->e_ival);
X              sub = copylist(sub,&tail);
X              if (back) {
X                   back->n_next = sub;
X                   tail->n_next = scan->n_next;
X              } else {
X                   head = sub;
X                   tail->n_next = scan->n_next;
X              }
X              scan->n_next = NULL;
X              freenode(scan);
X              scan = tail;
X         }
X    *rt = back;
X    return head;
X}
X
Xexpand(dp)
XNODEP dp;
X{
X    int nargs;
X    NODEP args;
X    register NODEP val;
X    NODEP tail;
X
X    val = dp->n_right;
X    if (val)
X         val = copylist(val, &tail);
X    nargs = dp->e_ival;
X    if (nargs >= 0) {
X         args = gath_args(nargs);
X         if (args) {
X              if (val)
X                   val = argfix(val,args,&tail);
X              freenode(args);
X         }
X    }
X    if (val == NULL)
X         return;
X#ifdef DEBUG
X    if (debugd > 1) {
X         printf("Expand");
X         printnode(val);
```

```
X    }
X#endif
X    tail->n_next = holdtok;
X    holdtok = val;
X}
X
Xp_undef()
X{
X    NODEP np, tp;
X
X    nxttok();
X    if (curtok.tnum != ID) {
X         error("bad #undef");
X         goto out;
X    }
X    tp = tok_to_node();
X    if ((np = hlook(deflist, tp)) != NULL)
X         /* quick and dirty */
X         np->n_name[0] = '#';
X    freenode(tp);
Xout:
X    ;
X}
X
X#endif
X#if PART_2
X
Xp_inc()
X{
X    int chkhere;
X    FILE *newf, *srch_open();
X    char *scopy(), *newnm;
X
X    tk_flags |= TK_NOESC|TK_LTSTR;
X    nxttok();
X    switch (curtok.tnum) {
X    case SCON:
X         chkhere = 1;
X         break;
X    case SCON2:
X         chkhere = 0;
X         break;
X    case NL:
X    case EOF:
X         error("bad #include");
X         return;
X    }
X    newf = srch_open(curstr, chkhere);
X    if (newf == NULL) {
X         fatals("Cant open ", curstr);
X         return;
X    }
X    newnm = scopy(curstr);
```

```
X    do
X         nxttok();
X    while (curtok.tnum != NL);
X    newfile(newf,newnm);
X}
X
Xint inclvl;
Xstruct svinc {
X    int lineno;
X    FILE *fd;
X    char *filenm;
X} svincs[MAXINCL];
X
X#if NEEDBUF
Xchar p_buf[MAXINCL][BUFSIZ];
X#endif
X
Xstatic char obuf[MAXSTR];
X
Xnewfile(fd,s)
XFILE *fd;
Xchar *s;
X{
X    register struct svinc *p;
X
X    inclvl++;
X    if (inclvl > MAXINCL) {
X         inclvl--;
X         fclose(fd);
X         error("too many includes");
X         return;
X    }
X    p = &svincs[inclvl-1];
X    p->lineno = lineno;
X    p->fd = input;
X    p->filenm = inname;
X    input = fd;
X    lineno = 1;
X    inname = s;
X#if NEEDBUF
X    setbuf(input, p_buf[inclvl-1]);
X#endif
X}
X
Xendfile()
X{
X    register struct svinc *p;
X
X    if (inclvl == 0)
X         return 0;
X    fclose(input);
X    inclvl--;
X    p = &svincs[inclvl];
```

```
X    sfree(inname);
X    input = p->fd;
X    lineno = p->lineno;
X    inname = p->filenm;
X    return 1;
X}
X
X#define MAXIDIR   10
X
X#ifndef FOR_AMIGA
X#ifndef MINIX
Xchar *srchlist[MAXIDIR] = {
X    "",
X    "\\include\\",
X    "\\sozobon\\include\\",
X    "",
X    0
X};
Xstatic int idir_n = 4;     /* number of entries in above table */
X#else
Xchar *srchlist[MAXIDIR] = {
X    "",
X    "/usr/include/",
X    0
X};
Xstatic int idir_n = 2;
X#endif
X#else
Xchar *srchlist[MAXIDIR] = {
X    "",
X    "include:",
X    0
X};
Xstatic int idir_n = 2;     /* number of entries in above table */
X#endif
X
Xstatic int idir_put = 1;   /* where to put -I dirs */
X
Xoptincl(s)
Xchar *s;
X{
X    register char **pp;
X
X    if (idir_n >= MAXIDIR-1) {
X         warn("too many -I dirs");
X         return;
X    }
X    for (pp = &srchlist[idir_n]; pp > &srchlist[idir_put]; ) {
X         pp--;
X         pp[1] = pp[0];
X    }
X    *pp = s;
X    idir_put++;
```

```
X    idir_n++;
X}
X
XFILE *
Xsrch_open(s, chkhere)
Xchar *s;
X{
X    char **dir;
X    FILE *fd;
X
X    dir = srchlist;
X    if (chkhere == 0) dir++;
X    while (*dir) {
X         strcpy(obuf, *dir);
X         strcat(obuf, s);
X         fd = fopen(obuf, ROPEN);
X         if (fd != NULL) {
X              return fd;
X         }
X         dir++;
X    }
X    return NULL;
X}
X
Xp_if(kind,skipping)
X{
X    int truth;
X    NODEP tp;
X
X    if (skipping) {
X         ifnest++;
X         return;
X    }
X    switch (kind) {
X    case 0:
X         truth = if_expr();
X         break;
X    case 1:
X    case 2:
X         nxttok();
X         if (curtok.tnum != ID) {
X              error("bad #if(n)def");
X              goto flush;
X         }
X         tp = tok_to_node();
X         truth = (hlook(deflist, tp) != NULL);
X         freenode(tp);
X         if (kind == 2)
X              truth = !truth;
X    }
X    iflevel++;
X    iftruth = truth;
Xflush:
```

```
X    ;
X}
X
Xif_expr()
X{
X    NODE *tp, *questx();
X    extern NODE *cur;
X    int rv;
X
X    in_if_x = 1;
X    skip_id = 0;
X    advnode();
X    tp = questx();
X    in_if_x = 0;
X    skip_id = 0;
X    if (tp) {
X         pnames(tp);
X         rv = conxval(tp);
X    } else
X         rv = 0;
X    if (cur->e_token != NL) {
X         error("bad #if");
X    } else
X         freenode(cur);
X    return rv;
X}
X
Xpnames(np)
Xregister NODEP np;
X{
Xagain:
X    if (np->e_token == ID) {
X         np->e_token = ICON;
X         np->e_ival = 0;
X    } else if (np->e_token == '(') {
X         NODEP rp, lp;
X         int truth;
X
X         rp = np->n_right;
X         lp = np->n_left;
X         if (lp->e_token == ID && strcmp(lp->n_name, "defined") == 0
X              && rp && rp->e_token == ID) {
X              truth = (hlook(deflist, rp) != NULL);
X              freenode(rp);
X              freenode(lp);
X              np->n_left = NULL;
X              np->n_right = NULL;
X              np->e_token = ICON;
X              np->e_ival = truth;
X         }
X    }
X    if (np->n_right)
X         pnames(np->n_right);
```

```
X    np = np->n_left;
X    if (np)
X         goto again;
X}
X
Xp_swit(kind,skipping)
X{
X    if (skipping && ifnest) {
X         if (kind == 1)
X              ifnest--;
X         return;
X    }
X    if (iflevel == 0) {
X         error("not in #if");
X         goto out;
X    }
X    switch (kind) {
X    case 0:       /* else */
X         iftruth = !iftruth;
X         break;
X    case 1:       /* endif */
X         iflevel--;
X         iftruth = 1;
X         break;
X    }
Xout:
X    ;
X}
X
Xp_line()
X{
X    char *scopy();
X
X    nxttok();
X    if (curtok.tnum != ICON) {
X         error("bad #line");
X         goto flush;
X    }
X    tk_flags |= TK_NOESC;
X    nxttok();
X    if (curtok.tnum == SCON) {
X         sfree(inname);
X         inname = scopy(curtok.name);
X    }
X    lineno = curtok.ival;
Xflush:
X    ;
X}
X
Xstruct cmds {
X    char *name;
X    int (*fun)();
X    int arg;
```

```
X    int skip;
X} pcmds[] = {
X    {"define", p_def, 0, 1},
X    {"undef", p_undef, 0, 1},
X    {"include", p_inc, 0, 1},
X    {"if", p_if, 0, 0},
X    {"ifdef", p_if, 1, 0},
X    {"ifndef", p_if, 2, 0},
X    {"else", p_swit, 0, 0},
X    {"endif", p_swit, 1, 0},
X    {"line", p_line, 0, 1},
X    {0}
X};
X
Xdopound(skipping)
X{
X    register struct cmds *p;
X    register char *cname;
X
X    tk_flags |= TK_SEENL;
X    sawnl = 0;
X    nxttok();
X    if (curtok.tnum != ID) {
X         error("expect name");
X         return;
X    }
X    cname = curtok.name;
X    for (p=pcmds; p->name; p++)
X         if (strcmp(p->name, cname) == 0) {
X              if (!skipping || !p->skip)
X                   (*p->fun)(p->arg, skipping);
X              tk_flags = 0;
X              if (sawnl == 0)
X                   tk_flags |= TK_ONLY1;
X              return;
X         }
X    error("bad # command");
X}
X
XNODEP
Xtok_to_node()
X{
X    register struct tok *tp;
X    register NODEP np;
X
X    tp = &curtok;
X    np = allocnode();
X    np->e_token = tp->tnum;
X    np->e_flags = tp->flags;
X    if (tp->prec) /* binary op */
X         np->e_prec = tp->prec;
X    else
X    switch (np->e_token) {
```

```
X    case ICON:
X         np->e_ival = tp->ival;
X         break;
X    case FCON:
X         np->e_fval = tp->fval;
X         break;
X    }
X    nscpy(np, tp->name);
X    return np;
X}
X
XNODEP
Xid_tok(s)
Xchar *s;
X{
X    NODEP np;
X
X    np = allocnode();
X    np->e_token = ID;
X    nscpy(np, s);
X    return np;
X}
X
X#endif
END_OF_FILE
if test 14315 -ne `wc -c <'hcc/PRE.C'`; then
    echo shar: \"'hcc/PRE.C'\" unpacked with wrong size!
fi
# end of 'hcc/PRE.C'
fi
echo shar: End of archive 7 \(of 9\).
cp /dev/null ark7isdone
MISSING=""
for I in 1 2 3 4 5 6 7 8 9 ; do
    if test ! -f ark${I}isdone ; then
     MISSING="${MISSING} ${I}"
    fi
done
if test "${MISSING}" = "" ; then
    echo You have unpacked all 9 archives.
    rm -f ark[1-9]isdone ark[1-9][0-9]isdone
else
    echo You still need to unpack the following archives:
    echo "        " ${MISSING}
fi
##  End of shell archive.
exit 0


From 4750964131190476867
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,2824e056239a1507,start
X-Google-Attributes: gid100493,public
```

X-Google-ArrivalTime: 1989-10-25 10:15:44 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!wuarchive!cs.utexas.edu!sun-barr!newstop!sun!sunfedcomm!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i099: sozobon1.2 -- Update to Sozobon C compiler part08/09
Keywords: shar, C
Message-ID: <34612@grapevine.uucp>
Date: 25 Oct 89 17:15:44 GMT
Sender: daemon@sunfedcomm
Reply-To: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Lines: 1742
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Posting-number: Volume 2, Issue 99
Archive-name: sozobon1.2/part08

```
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of archive 8 (of 9)."
# Contents:  top/NPEEP2.C top/REG.C
# Wrapped by koreth@panarthea on Tue Oct 24 18:40:47 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'top/NPEEP2.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/NPEEP2.C'\"
else
echo shar: Extracting \"'top/NPEEP2.C'\" \(17335 characters\)
sed "s/^X//" >'top/NPEEP2.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X
X/*
X * 2-instruction peephole optimizations
X */
X
X#include "top.h"
X
X/*
X * Macros to reference commonly-used values... cleans up the following
X * code quite a bit.
```

```
X */
X#define  sm1  i1->src.amode /* source & dest addressing modes */
X#define  dm1  i1->dst.amode
X#define  sm2  i2->src.amode
X#define  dm2  i2->dst.amode
X
X#define  sr1  i1->src.areg  /* source & dest registers */
X#define  dr1  i1->dst.areg
X#define  sr2  i2->src.areg
X#define  dr2  i2->dst.areg
X
X
X/*
X * ipeep2(bp, i1) - look for 2-instruction optimizations at the given inst.
X */
Xstatic  int
Xipeep2(bp, i1)
XBLOCK   *bp;
Xregister INST    *i1;
X{
X    register INST *i2;     /* the next instruction */
X    register INST *ti2;         /* "temporary" next inst */
X
X    register int  op1, op2;/* opcodes, for speed */
X
X    i2  = i1->next;
X    op1 = i1->opcode;
X    op2 = i2->opcode;
X
X    /*
X     * Avoid stack fix-ups after a call if possible.
X     */
X
X    /*
X     *   addq #4,sp
X     *   ... stuff that doesn't use SP ...
X     *   move.l   ?,-(sp)       =>   move.l   ?,(sp)
X     */
X    if (op1 == ADDQ && sm1 == IMM && i1->src.disp == 4 &&
X        dm1 == REG && dr1 == SP) {
X
X        ti2 = i2;
X        while (!uses(ti2, SP)) {
X             if (ti2->next == NULL)
X                  goto end2;
X             ti2 = ti2->next;
X        }
X
X        if (ti2->opcode == MOVE && ti2->flags == LENL &&
X            ti2->dst.amode == (REGI|DEC) && ti2->dst.areg == SP) {
X             ti2->dst.amode = REGI;
X             delinst(bp, i1);
X             DBG(printf("%d ", __LINE__))
```

```
X             return CLEAN;
X         }
X    }
Xend2:
X
X    /*
X     *   addq #2,sp
X     *   ... stuff that doesn't use SP ...
X     *   move.w    ?,-(sp)       =>   move.w    ?,(sp)
X     */
X    if (op1 == ADDQ && sm1 == IMM && i1->src.disp == 2 &&
X        dm1 == REG && dr1 == SP) {
X
X         ti2 = i2;
X         while (!uses(ti2, SP)) {
X              if (ti2->next == NULL)
X                   goto end3;
X              ti2 = ti2->next;
X         }
X
X         if (ti2->opcode == MOVE && ti2->flags == LENW &&
X             ti2->dst.amode == (REGI|DEC) && ti2->dst.areg == SP) {
X              ti2->dst.amode = REGI;
X              delinst(bp, i1);
X              DBG(printf("%d ", __LINE__))
X              return CLEAN;
X         }
X    }
Xend3:
X
X    /*
X     * Avoid "tst" instructions following instructions that
X     * set the Z flag.
X     */
X
X    /*
X     *   move.x    X, Y      =>   move.x    X, Y
X     *   tst.x     X or Y             ...deleted...
X     *   beq/bne                 beq/bne
X     *
X     *   Where Y is not An, because "movea" doesn't set the
X     *   zero flag.
X     */
X    if (bp->last == i2 && (bp->bcode == BEQ || bp->bcode == BNE) &&
X        op1 == MOVE && op2 == TST &&
X        i1->flags == i2->flags) {
X
X         /*
X          * If pre-decrement is set on the dest. of the move,
X          * don't let that screw up the operand comparison.
X          */
X         if (dm1 & DEC)
X              dm1 &= ~DEC;
```

```
X
X         if (opeq(&i1->dst, &i2->src) || opeq(&i1->src, &i2->src)) {
X              if (dm1 != REG || ISD(dr1)) {
X                   delinst(bp, i2);
X                   DBG(printf("%d ", __LINE__))
X                   return CLEAN;
X              }
X         }
X    }
X
X    /*
X     *   and.x     X, Y      =>   and.x     X, Y
X     *   tst.x     X or Y              ...deleted...
X     *   beq/bne                beq/bne
X     *
X     *   Where Y is not An, because "movea" doesn't set the
X     *   zero flag.
X     */
X    if (bp->last == i2 && (bp->bcode == BEQ || bp->bcode == BNE) &&
X        op1 == AND && op2 == TST &&
X        i1->flags == i2->flags) {
X
X         /*
X          * If pre-decrement is set on the dest. of the move,
X          * don't let that screw up the operand comparison.
X          */
X         if (dm1 & DEC)
X              dm1 &= ~DEC;
X
X         if (opeq(&i1->dst, &i2->src) || opeq(&i1->src, &i2->src)) {
X              if (dm1 != REG || ISD(dr1)) {
X                   delinst(bp, i2);
X                   DBG(printf("%d ", __LINE__))
X                   return CLEAN;
X              }
X         }
X    }
X
X    /*
X     *   ext.x     Dn        =>   ext.x     Dn
X     *   tst.x     Dn             ...deleted...
X     *   beq/bne                beq/bne
X     *
X     *   Where Y is not An, because "movea" doesn't set the
X     *   zero flag.
X     */
X    if ((bp->last == i2) && (bp->bcode == BEQ || bp->bcode == BNE) &&
X        (op1 == EXT) && (op2 == TST) &&
X        (i1->flags == i2->flags)) {
X
X         if ((sm1 == REG) && ISD(sr1) && (sm2 == REG) && (sr1 == sr2)) {
X              delinst(bp, i2);
X              DBG(printf("%d ", __LINE__))
```

```
X             return CLEAN;
X         }
X    }
X
X    /*
X     *   move.?    X, Dn         =>   move.?    X, Dn
X     *   ext.?     Dn            ...deleted...
X     *   beq/bne                 beq/bne
X     *
X     *   Where Dn is dead after the "ext".
X     */
X    if (bp->last == i2 && (bp->bcode == BEQ || bp->bcode == BNE) &&
X        op1 == MOVE && op2 == EXT) {
X
X         if ((dm1 == REG) && ISD(dr1) &&
X             (sm2 == REG) && (dr1 == sr2)) {
X              if ((i2->live & RM(sr2)) == 0) {
X                   delinst(bp, i2);
X                   DBG(printf("%d ", __LINE__))
X                   return CLEAN;
X              }
X         }
X    }
X
X    /*
X     *   ext.l     Dm       =>   ...deleted...
X     *   tst.l     Dm            tst.w     Dm
X     *
X     *   where Dm is dead after the "tst".
X     */
X    if (op1 == EXT && op2 == TST &&
X        ((i1->flags & LENL) != 0) &&  ((i2->flags & LENL) != 0) &&
X        (sr1 == sr2) && ISD(sr1)) {
X
X         if ((i2->live & RM(sr2)) == 0) {
X              i2->flags = LENW;
X              delinst(bp, i1);
X              DBG(printf("%d ", __LINE__))
X              return CLEAN;
X         }
X    }
X
X#if 0
X    /*
X     *   add[q]    #?,sp
X     *   ... stuff that doesn't use SP ...
X     *   unlk An        =>   unlk An
X     */
X    if ((op1 == ADDQ || op1 == ADD) && sm1 == IMM &&
X         dm1 == REG && dr1 == SP) {
X
X         ti2 = i2;
X         while (!uses(ti2, SP)) {
```

```
X             if (ti2->next == NULL)
X                  goto end8;
X             ti2 = ti2->next;
X        }
X        if (ti2->opcode == UNLK) {
X                  delinst(bp, i1);
X             DBG(printf("%d ", __LINE__))
X                  return CLEAN;
X        }
X    }
Xend8:
X#endif
X
X    /*
X     *   ext.l     Dm        =>   ...deleted...
X     *   ???  N(An,Dm.l), ??          ???  N(An,Dm.w), ??
X     *
X     *   Where Dm is dead
X     */
X    if ((op1 == EXT) && (i1->flags & LENL) &&
X        (sm2 == (REGIDX|XLONG)) &&
X        (sr1 == i2->src.ireg)) {
X
X        if ((i2->live & RM(sr1)) == 0) {
X             sm2 &= ~XLONG;
X             delinst(bp, i1);
X             DBG(printf("%d ", __LINE__))
X             return CLEAN;
X        }
X    }
X
X    /*
X     *   ext.l     Dm        =>   ...deleted...
X     *   ???  ??, N(An,Dm.l)          ???  ??, N(An,Dm.w)
X     *
X     *   Where Dm is dead
X     */
X    if ((op1 == EXT) && (i1->flags & LENL) &&
X        (dm2 == (REGIDX|XLONG)) &&
X        (sr1 == i2->dst.ireg)) {
X
X        if ((i2->live & RM(sr1)) == 0) {
X             dm2  &= ~XLONG;
X             delinst(bp, i1);
X             DBG(printf("%d ", __LINE__))
X             return CLEAN;
X        }
X    }
X
X    /*
X     * Avoid intermediate registers.
X     */
X
```

```
X    /*
X     *   move.x    X, Dm         =>   INST.x    X, Dn
X     *   INST.x    Dm, Dn
X     *
X     *   where Dm is dead, and INST is one of: add, sub, and, or, cmp
X     */
X    if ((op1 == MOVE) &&
X        ((op2==ADD)||(op2==SUB)||(op2==AND)||(op2==OR)||(op2==CMP)) &&
X        (i1->flags == i2->flags) &&
X        (dm1 == REG) && ISD(dr1) &&
X        (sm2 == REG) && ISD(sr2) &&
X        (dm2 == REG) && ISD(dr2) &&
X        (dr1 == sr2)) {
X
X         if ((i2->live & RM(sr2)) == 0) {
X
X              i1->opcode = i2->opcode;
X              dr1 = dr2;
X
X              delinst(bp, i2);
X              DBG(printf("%d ", __LINE__))
X              return DIRTY;
X         }
X    }
X
X    /*
X     * Silly moves
X     */
X
X    /*
X     *   move.x    X, Y      =>   move.x    X, Y
X     *   move.x    Y, X
X     */
X    if ((op1 == MOVE) && (op2 == MOVE) &&
X        (i1->flags == i2->flags) &&
X        opeq(&i1->src, &i2->dst) && opeq(&i1->dst, &i2->src) &&
X        ((sm1 & (INC|DEC)) == 0) &&
X        ((dm1 & (INC|DEC)) == 0)) {
X
X              delinst(bp, i2);
X         DBG(printf("%d ", __LINE__))
X              return CLEAN;
X    }
X
X    /*
X     *   move.x    X, Y      =>   move.x    X, Rn
X     *   move.x    Y, Rn              move.x    Rn, Y
X     *
X     *   where Y isn't INC or DEC, and isn't register direct
X     */
X    if ((op1 == MOVE) && (op2 == MOVE) && (dm2 == REG) &&
X        opeq(&i1->dst, &i2->src) && ((dm1 & (INC|DEC)) == 0) &&
X        (i1->flags == i2->flags) && (dm1 != REG)) {
```

```
X
X        freeop(&i1->dst);
X        i1->dst = i2->dst;
X        i2->dst = i2->src;
X        i2->src = i1->dst;
X
X        DBG(printf("%d ", __LINE__))
X        return CLEAN;
X    }
X
X    /*
X     *   move.x    Dm, X         =>   move.x    Dm, X
X     *   move.x    X, Y          move.x    Dm, Y
X     *
X     * Where 'x' is the same, and 'X' has no side-effects.
X     */
X    if ((op1 == MOVE) && (op2 == MOVE) &&
X        (sm1 == REG) && ISD(sr1) &&
X        (i1->flags == i2->flags) && opeq(&i1->dst, &i2->src) &&
X        ((dm1 & (DEC|INC)) == 0)) {
X
X
X        freeop(&i2->src);
X        i2->src = i1->src;
X        DBG(printf("%d ", __LINE__))
X             return DIRTY;
X    }
X
X#if 0
X    /*
X     *   move.?    Rm, Rn        =>   move.?    Rm, Rn
X     *   ... stuff ...           ... stuff ...
X     *   move.?    Rm, Rn
X     *
X     *   where "stuff" doesn't set Rm or Rn. Also make sure a
X     *   conditional branch doesn't directly follow the second
X     *   move. If so, we need to leave the move alone since it
X     *   sets condition codes.
X     */
X    if ((op1 == MOVE) && (sm1 == REG) && (dm1 == REG)) {
X        ti2 = i2;
X        while (ti2 != NULL && !sets(ti2, sr1) && !sets(ti2, dr1)) {
X
X             if ((ti2->opcode==MOVE) && (i1->flags==ti2->flags) &&
X                 (ti2->src.amode==REG) && (ti2->dst.amode==REG) &&
X                 (sr1 == ti2->src.areg) &&
X                 (dr1 == ti2->dst.areg)) {
X
X                      delinst(bp, ti2);
X                  DBG(printf("%d ", __LINE__))
X                      return DIRTY;
X             }
X             ti2 = ti2->next;
```

```
X        }
X    }
X#endif
X
X    /*
X     *   move.l    Am, Dn        =>   move.l    Am, Ao
X     *   ... stuff ...           ... stuff ...
X     *   move.l    Dn, Ao
X     *
X     *   where "stuff" doesn't set Dn.
X     */
X    if ((op1 == MOVE) && (i1->flags == LENL) &&
X        (sm1 == REG) && ISA(sr1) && (dm1 == REG) && ISD(dr1)) {
X
X        ti2 = i2;
X        while (!sets(ti2, dr1)) {
X
X            if ((ti2->opcode == MOVE) && (ti2->flags == LENL) &&
X                (ti2->src.amode == REG) && ISD(ti2->src.areg) &&
X                (ti2->dst.amode == REG) && ISA(ti2->dst.areg) &&
X                (dr1 == ti2->src.areg)) {
X
X                /*
X                 * If the intermediate register isn't dead,
X                 * then we have to keep using it.
X                 */
X                if ((ti2->live & RM(ti2->src.areg)) != 0)
X                    goto end14;
X
X                dr1 = ti2->dst.areg;
X
X                    delinst(bp, ti2);
X                DBG(printf("%d ", __LINE__))
X                    return DIRTY;
X            }
X
X            if (ti2->next == NULL)
X                goto end14;
X
X            ti2 = ti2->next;
X        }
X    }
Xend14:
X
X    /*
X     *   move.l    Dm, An        =>   move.l    Dm, Ao
X     *   lea  (An), Ao
X     *
X     *   where An is dead
X     */
X    if ((op1 == MOVE) && (op2 == LEA) && (sm1 == REG) && ISD(sr1) &&
X        (dm1 == REG) && ISA(dr1) && (sm2 == REGI) && (dm2 == REG) &&
X        ISA(dr2) && (dr1 == sr2)) {
```

```
X
X        if ((i2->live & RM(sr2)) == 0) {
X
X             dr1 = dr2;
X             i1->live = i2->live;
X
X             delinst(bp, i2);
X             DBG(printf("%d ", __LINE__))
X             return CLEAN;
X        }
X    }
X
X    /*
X     *   lea  X, An         =>   lea  X, Ao
X     *   lea  (An), Ao
X     *
X     *   where An is dead
X     */
X    if ((op1 == LEA) && (op2 == LEA) && (sm2 == REGI) && (dr1 == sr2)) {
X
X        if ((i2->live & RM(sr2)) == 0) {
X
X             dr1 = dr2;
X             i1->live = i2->live;
X
X             delinst(bp, i2);
X             DBG(printf("%d ", __LINE__))
X             return CLEAN;
X        }
X    }
X
X    /*
X     *   lea  N(Am), Am=>
X     *   ?    (Am)[,...]         ?    N(Am)[,...]
X     *
X     *   Where Am is either dead after the second instruction or
X     *   is a direct destination of the second instruction.
X     */
X    if ((op1 == LEA) && (sm1 == REGID) && (sr1 == dr1) &&
X        (sm2 == REGI) && (dr1 == sr2)) {
X
X        if (((i2->live & RM(sr2)) == 0) ||
X           ((dm2 == REG) && (dr2 == sr2))) {
X             i2->src.amode = REGID;
X             i2->src.disp = i1->src.disp;
X             delinst(bp, i1);
X             DBG(printf("%d ", __LINE__))
X             return DIRTY;
X        }
X    }
X
X    /*
X     *   lea  N(Am), Am=>
```

```
X     *   ?    X, (Am)            ?    X, N(Am)
X     *
X     *   Where X doesn't reference Am, and Am is dead after the
X     *   second instruction.
X     */
X    if ((op1 == LEA) && (sm1 == REGID) &&
X        (sr1 == dr1) && (dm2 == REGI) && (dr1 == dr2)) {
X
X        if (((i2->live & RM(dr2)) == 0) &&
X            ((sm2 == IMM) || (sm2 == ABS) || (sr2 != dr2))) {
X            dm2 = REGID;
X            i2->dst.disp = i1->src.disp;
X            delinst(bp, i1);
X            DBG(printf("%d ", __LINE__))
X            return DIRTY;
X        }
X    }
X
X    /*
X     *   lea  X, Am         =>   ...deleted...
X     *   clr.x    (Am)          clr.x    X
X     *
X     *   where Am is dead
X     */
X    if ((op1 == LEA) && (op2 == CLR) && (sm2 == REGI) && (dr1 == sr2)) {
X
X        if ((i2->live & RM(sr2)) == 0) {
X            i2->src = i1->src;
X
X            delinst(bp, i1);
X            DBG(printf("%d ", __LINE__))
X            return CLEAN;
X        }
X    }
X
X    /*
X     *   lea  X, Am         =>   ...deleted...
X     *   move.x   Y, (Am)           move.x   Y, X
X     *
X     *   where Am is dead
X     */
X    if ((op1 == LEA) && (op2 == MOVE) && dm2 == REGI && (dr1 == dr2)) {
X
X        if ((i2->live & RM(dr2)) == 0) {
X            i2->dst = i1->src;
X
X            delinst(bp, i1);
X            DBG(printf("%d ", __LINE__))
X            return CLEAN;
X        }
X    }
X
X    /*
```

```
X     *   lea  X, Am         =>   ...deleted...
X     *   move.x    (Am), Y           move.x    X, Y
X     *
X     *   where Am is dead
X     */
X    if ((op1 == LEA) && (op2 == MOVE) && sm2 == REGI && (dr1 == sr2)) {
X
X        if ((i2->live & RM(sr2)) == 0) {
X             i2->src = i1->src;
X
X             delinst(bp, i1);
X             DBG(printf("%d ", __LINE__))
X             return CLEAN;
X        }
X    }
X
X    /*
X     *   move.x    Dm, X         =>   move.x    Dm, X
X     *   cmp.x     #N, X              cmp.x     #N, Dm
X     *
X     *   Where X isn't register direct.
X     *
X     *   Since X generally references memory, we can compare
X     *   with the register faster.
X     */
X    if ((op1 == MOVE) && (op2 == CMP) &&
X        (i1->flags == i2->flags) && (sm2 == IMM) &&
X        (sm1 == REG) && ISD(sr1) && (dm1 != REG) &&
X        opeq(&i1->dst, &i2->dst) && ((dm1 & (INC|DEC)) == 0)) {
X
X        freeop(&i2->dst);
X        dm2 = REG;
X        dr2 = sr1;
X
X        DBG(printf("%d ", __LINE__))
X        return DIRTY;
X    }
X
X
X    /*
X     * Try to use register indirect w/ displacement and/or index
X     */
X
X    /*
X     *   add.l     Am, Dn        =>   lea  0(Am,Dn.l), Ao
X     *   move.l    Dn, Ao
X     *
X     *   where Dn is dead
X     */
X    if ((op1 == ADD) && (op2 == MOVE) &&
X        (sm1 == REG) && ISA(sr1) &&
X        (dm1 == REG) && ISD(dr1) &&
X        (sm2 == REG) && ISD(sr2) &&
```

```
X        (dm2 == REG) && ISA(dr2) &&
X        (dr1 == sr2) &&
X        (i1->flags & LENL) && (i2->flags & LENL)) {
X
X        if ((i2->live & RM(sr2)) == 0) {
X
X             i2->opcode = LEA;
X             i2->flags = 0;
X
X             sm2 = REGIDX|XLONG;
X             i2->src.disp = 0;
X             sr2 = sr1;
X             i2->src.ireg = dr1;
X
X             delinst(bp, i1);
X             DBG(printf("%d ", __LINE__))
X             return CLEAN;
X        }
X    }
X
X    /*
X     *   add.l    Dm, An        =>   move.x   0(An,Dm.l), Do
X     *   move.x   (An), Do
X     *
X     *   where An is dead
X     */
X    if ((op1 == ADD) && (op2 == MOVE) &&
X        (sm1 == REG) && ISD(i1->src.areg) &&
X        (dm1 == REG) && ISA(dr1) &&
X        (sm2 == REGI)&& ISA(sr2) &&
X        (dm2 == REG) && ISD(dr2) &&
X        (dr1 == sr2) && (i1->flags & LENL)) {
X
X        if ((i2->live & RM(sr2)) == 0) {
X
X             sm2 = REGIDX|XLONG;
X             i2->src.disp = 0;
X             i2->src.ireg = sr1;
X
X             delinst(bp, i1);
X             DBG(printf("%d ", __LINE__))
X             return CLEAN;
X        }
X    }
X
X    /*
X     *   lea  N(Am), An =>   lea  N(Am,Do.l), An
X     *   add.l    Do, An
X     *
X     */
X    if ((op1 == LEA) && (op2 == ADD) && (sm1 == REGID) &&
X        (sm2 == REG) && ISD(sr2) && (dm2 == REG) && ISA(dr2) &&
X        (dr1 == dr2) && D8OK(i1->src.disp)) {
```

```
X
X         sm1 = REGIDX|XLONG;
X         i1->src.ireg = sr2;
X         i1->live = i2->live;
X         delinst(bp, i2);
X         DBG(printf("%d ", __LINE__))
X         return CLEAN;
X    }
X
X
X
X    /*
X     * Try to use the pre-decrement and post-increment modes
X     * whenever possible.
X     */
X
X    /*
X     *   sub.l    #1, Am
X     *   ... stuff ...
X     *   ???.b    ..(Am).. =>   ???.b    ..-(Am)..
X     *
X     *   Nothing in "stuff" can refer to Am.
X     */
X    if ((op1 == SUBQ) && (i1->flags & LENL) &&
X        (sm1 == IMM) && (i1->src.disp == 1) &&
X        (dm1 == REG) && ISA(dr1)) {
X
X         while (i2 != NULL) {
X
X              if (sm2 == REGI && sr2 == dr1) {
X
X                   if ((i2->flags & LENB) == 0)
X                        goto end24;
X
X                   sm2 |= DEC;
X
X                   delinst(bp, i1);
X                   DBG(printf("%d ", __LINE__))
X                   return DIRTY;
X              }
X              if (dm2 == REGI && dr2 == dr1) {
X
X                   if ((i2->flags & LENB) == 0)
X                        goto end24;
X
X                   dm2 |= DEC;
X
X                   delinst(bp, i1);
X                   DBG(printf("%d ", __LINE__))
X                   return CLEAN;
X              }
X
X              if (uses(i2, RM(dr1)))
```

```
X                 goto end24;
X
X             if (i2->next == NULL)
X                 goto end24;
X             else
X                 i2 = i2->next;
X
X         }
X    }
Xend24:
X
X    /*
X     *   sub.l    #2, Am
X     *   ... stuff ...
X     *   ???.w    ..(Am).. =>   ???.w    ..-(Am)..
X     *
X     *   Nothing in "stuff" can refer to Am.
X     */
X    if ((op1 == SUBQ) && (i1->flags & LENL) &&
X        (sm1 == IMM) && (i1->src.disp == 2) &&
X        (dm1 == REG) && ISA(dr1)) {
X
X        while (i2 != NULL) {
X
X             if (sm2 == REGI && sr2 == dr1) {
X
X                 if ((i2->flags & LENW) == 0)
X                     goto end26;
X
X                 sm2 |= DEC;
X
X                 delinst(bp, i1);
X                 DBG(printf("%d ", __LINE__))
X                 return CLEAN;
X             }
X             if (dm2 == REGI && dr2 == dr1) {
X
X                 if ((i2->flags & LENW) == 0)
X                     goto end26;
X
X                 dm2 |= DEC;
X
X                 delinst(bp, i1);
X                 DBG(printf("%d ", __LINE__))
X                 return CLEAN;
X             }
X
X             if (uses(i2, RM(dr1)))
X                 goto end26;
X
X             if (i2->next == NULL)
X                 goto end26;
X             else
```

```
X                  i2 = i2->next;
X
X         }
X    }
Xend26:
X
X    /*
X     *   sub.l    #4, Am
X     *   ... stuff ...
X     *   ???.l    ..(Am).. =>   ???.l    ..-(Am)..
X     *
X     *   Nothing in "stuff" can refer to Am.
X     */
X    if ((op1 == SUBQ) && (i1->flags & LENL) &&
X        (sm1 == IMM) && (i1->src.disp == 4) &&
X        (dm1 == REG) && ISA(dr1)) {
X
X         while (i2 != NULL) {
X
X              if (sm2 == REGI && sr2 == dr1) {
X
X                   if ((i2->flags & LENL) == 0)
X                        goto end28;
X
X                   sm2 |= DEC;
X
X                   delinst(bp, i1);
X                   DBG(printf("%d ", __LINE__))
X                   return CLEAN;
X              }
X              if (dm2 == REGI && dr2 == dr1) {
X
X                   if ((i2->flags & LENL) == 0)
X                        goto end28;
X
X                   i2->dst.amode |= DEC;
X
X                   delinst(bp, i1);
X                   DBG(printf("%d ", __LINE__))
X                   return CLEAN;
X              }
X
X              if (uses(i2, RM(dr1)))
X                   goto end28;
X
X              if (i2->next == NULL)
X                   goto end28;
X              else
X                   i2 = i2->next;
X
X         }
X    }
Xend28:
```

```
X
X    return UNCHANGED;
X}
X
X/*
X * peep2(bp) - scan blocks starting at 'bp'
X */
Xint
Xpeep2(bp)
Xregister BLOCK    *bp;
X{
X    register INST *ip;
X    register bool i;
X    register int  state = UNCHANGED;
X
X    DBG(printf("p2 :"))
X
X    for (; bp != NULL ;bp = bp->next) {
X         for (ip = bp->first; ip != NULL && ip->next != NULL ;) {
X              if ((i = ipeep2(bp, ip)) != UNCHANGED) {
X
X                   if (i == DIRTY)
X                        state = DIRTY;
X                   if (state == UNCHANGED)
X                        state = CLEAN;
X
X                   s_peep2++;
X                   /*
X                    * If we had a match, then either instruction
X                    * may have been deleted, so the safe thing to
X                    * do is to go to the next block.
X                    */
X                   break;
X              } else
X                   ip = ip->next;
X         }
X    }
X    DBG(printf("\n"); fflush(stdout))
X    return state;
X}
END_OF_FILE
if test 17335 -ne `wc -c <'top/NPEEP2.C'`; then
    echo shar: \"'top/NPEEP2.C'\" unpacked with wrong size!
fi
# end of 'top/NPEEP2.C'
fi
if test -f 'top/REG.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/REG.C'\"
else
echo shar: Extracting \"'top/REG.C'\" \(17767 characters\)
sed "s/^X//" >'top/REG.C' <<'END_OF_FILE'
X/* Copyright (c) 1989 by Sozobon, Limited.  Author: Tony Andrews
X *
```

```
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X
X/*
X * The code in this file deals with "registerizing" local variables and
X * parameters. The general idea is to look for highly referenced local
X * variables and parameters and effectively turn them into register
X * variables automatically. Only the D registers are used, currently, so
X * for pointer variables, a manual "register" declaration in the source
X * code is actually better.
X *
X * We need to be certain of several things about a variable before placing
X * it in a register. It's address must not be taken, and it must not be
X * referred to through "aliases" (e.g. when casting to a shorter object).
X * It must be able to fit in a register. And to keep things like printf from
X * breaking, parameters can only be registerized if none of the parameters
X * have their address taken.
X *
X * The compiler makes this all possible by placing "hints" within the
X * generated assembly code. These hints appear as comments, but are parsed
X * by the optimizer, and the information is stashed away by calling addvar().
X * The hints give us the size and offset of each parameter and local variable.
X * Their names are also given, although that information isn't needed here.
X *
X * There are tradeoffs to be wary of when registerizing. If no register
X * variables exist yet, then "movem" instructions have to be added, requiring
X * more references to make this worthwhile. In the case of parameters, the
X * register has to be initialized from the stack. The four cases are:
X *
X *  Locals   w/ other regs:     1 reference  required
X *       no other regs:     4 references required
X *  Parms    w/ other regs:     2 references required
X *       no other regs:     6 references required
X *
X * The numbers above represent the break-even point based on a savings of
X * 2 bytes per reference, and the incremental cost of adding "movem" or
X * "move" instructions as needed.
X *
X * This optimizes for space only. To optimize for time, each reference would
X * be weighted based on the loop nesting level at which it occurs.
X */
X
X#include "top.h"
X
X#define  MAXLOCALS 100
X
Xstatic   struct   linfo {
```

```
X    long offset;       /* offset from A6 */
X    int  size;         /* size of the object */
X    int  ref;      /* # of references to the local */
X    int  reg;      /* reg. we assigned it to */
X    int  flags;        /* length, etc. */
X} locals[MAXLOCALS];
X
X#define  ALIASED       0x1  /* offset is aliased with another */
X#define  ADDR_TAKEN    0x2  /* address of the variable was taken */
X
X#define  IS_LOCAL(x)   (locals[(x)].offset < 0)
X#define  IS_PARM(x)    (locals[(x)].offset > 0)
X
Xstatic   bool paddr;        /* address of a parameter was taken */
Xstatic   int  lcnt;         /* number of local variables we've seen */
Xstatic   int  rcnt;         /* number of locals that got registerized */
X
Xstatic   int  omask, nmask; /* old and new register masks */
X
X/*
X * addvar(size, off) - add a variable entry for the current function
X *
X * These come from hints the compiler gives us about local variables.
X * We use the size and offset here to make sure we don't have aliasing
X * problems with the local variables we want to registerize.
X */
Xvoid
Xaddvar(size, off)
Xint size;
Xint off;
X{
X    locals[lcnt].offset = off;
X    locals[lcnt].size = size;
X    locals[lcnt].flags = 0;
X    locals[lcnt].ref = 0;
X
X    lcnt++;
X}
X
X/*
X * clrvar() - clear the variable list
X */
Xvoid
Xclrvar()
X{
X    register int  i;
X
X    /*
X     * re-initialize the local information
X     */
X    for (i=0; i < MAXLOCALS ;i++) {
X        locals[i].ref = 0;
X        locals[i].reg = -1;
```

```
X        locals[i].flags = 0;
X        locals[i].offset = 0;
X        locals[i].size = 0;
X    }
X    paddr = FALSE;
X    rcnt = lcnt = 0;
X}
X
X/*
X * setreg() - try to "registerize" local variables in the given function
X */
Xvoid
Xsetreg(bp)
XBLOCK   *bp;
X{
X    void lcheck(), lassign(), lrewrite();
X
X    lcheck(bp);
X    lassign();
X
X#ifdef  DEBUG
X    if (debug)
X        dump_table();
X#endif
X
X    if (rcnt > 0)
X        lrewrite(bp);
X
X    s_reg += rcnt;          /* keep totals for accounting */
X}
X
X/*
X * lcheck() - scan for local variable references in the given function
X */
Xstatic void
Xlcheck(bp)
XBLOCK   *bp;
X{
X    void ckref();
X    register int  i;
X    register BLOCK    *cb;
X    register INST *ci;
X
X    for (cb = bp; cb != NULL ;cb = cb->next) {
X        for (ci = cb->first; ci != NULL ;ci = ci->next) {
X            ckref(ci, &ci->src);
X            ckref(ci, &ci->dst);
X        }
X    }
X
X    /*
X     * Now figure out which registers are currently used.
X     */
```

```
X    ci = bp->first->next;
X
X    if (ci != NULL && ci->opcode == MOVEM) {
X         if (ci->src.amode == REG)
X              omask = RM(ci->src.areg);
X         else
X              omask = stomask(ci->src.astr);
X    } else
X         omask = 0;
X}
X
X/*
X * ckref() - check for a local variable reference
X *
X * If a local variable reference is found, it's added to the table or
X * (if already there) its reference count is incremented. If we're
X * taking its address, note that too.
X */
Xstatic void
Xckref(ip, op)
XINST    *ip;
Xstruct  opnd *op;
X{
X    register int  i;
X    register int  sz;
X
X    if (op->amode != REGID || op->areg != A6)
X         return;
X
X    switch (ip->flags) {
X    case LENL:
X         sz = 4;
X         break;
X    case LENW:
X         sz = 2;
X         break;
X    case LENB:
X    default:      /* for LEA and PEA */
X         sz = 1;
X         break;
X    }
X
X    /*
X     * is the local variable already in the table?
X     */
X    for (i=0; i < lcnt ;i++) {
X         if (locals[i].offset == op->disp && locals[i].size == sz) {
X              locals[i].ref++;
X              break;
X         }
X    }
X
X    /*
```

```
X     * If not in the table, add an entry for it. If we add an entry
X     * here, it must be an alias for one of the entries we got via
X     * the compiler hints.
X     */
X    if (i == lcnt) {
X         locals[lcnt].offset = op->disp;
X         locals[lcnt].size = sz;
X         locals[lcnt].flags = 0;
X         locals[lcnt].ref = 1;
X
X         lcnt++;
X    }
X
X    if (ip->opcode == LEA || ip->opcode == PEA) {
X         locals[i].flags = ADDR_TAKEN;
X         /*
X          * If we took the address of a parameter, note that
X          * by setting 'paddr'.
X          */
X         if (IS_PARM(i))
X              paddr = TRUE;
X    }
X}
X
X/*
X * lassign() - assign local variable to registers
X *
X * Check for aliases, sort the table, and then decide how to assign
X * the local variables to registers.
X */
Xstatic void
Xlassign()
X{
X    void ck_aliases(), sort_table(), do_sort();
X    register int  i;
X    register int  r;
X    int  minlref; /* min. required references for a local */
X    int  minpref; /* min. required references for a parameter */
X
X    ck_aliases();      /* disqualify any "aliased" references */
X    sort_table();      /* and sort by reference count */
X
X    /*
X     * If there were already "movem" instructions, then we should
X     * convert as many locals as possible to registers. If we're
X     * going to have to add the movem's, then we need at least 4
X     * references for this to be worthwhile. The 2 movem instructions
X     * take 8 bytes, and each reference conversion saves 2 bytes.
X     * This analysis optimizes for size.
X     */
X    minlref = (omask != 0) ? 1 : 4;
X    minpref = (omask != 0) ? 2 : 6;
X
```

```
X    nmask = omask;
X
X    for (i=0, r=D3; r <= D7 ;) {
X
X         /*
X          * If the register is already in use, skip it.
X          */
X         if (omask & RM(r)) {
X              r++;
X              continue;
X         }
X
X         /*
X          * If no more eligible variables, then stop.
X          */
X         if (locals[i].ref <= 0)
X              break;
X
X         /*
X          * If something meets the minimums, then assign it to
X          * the current register, and adjust the minimums.
X          */
X         if ((IS_LOCAL(i) && locals[i].ref >= minlref) ||
X             (IS_PARM(i)  && locals[i].ref >= minpref)) {
X              locals[i].reg = r;
X              nmask |= RM(r);
X              minlref = 1;
X              minpref = 2;
X              r++;
X              i++;
X         } else {
X              /*
X               * If we run into something that isn't referenced
X               * enough, disqualify it and re-sort. There might
X               * still be something else worth doing.
X               */
X              locals[i].ref = -locals[i].ref;
X              do_sort();
X         }
X    }
X    rcnt = i;
X}
X
X/*
X * ck_aliases() - check for aliases in the locals table
X *
X * An alias occurs when two different offsets off of A6 both reference
X * the same local. This can happen when casting to a smaller type. Since
X * these references would be a pain to rewrite, we just bag it.
X */
Xstatic void
Xck_aliases()
X{
```

```
X    static   bool ck_aref();
X    register int  i;
X    register int  s;
X    register long d;
X
X    for (i=0; i < lcnt ;i++) {
X         d = locals[i].offset;
X         s = locals[i].size;
X
X         if (ck_aref(d, s))
X              locals[i].flags |= ALIASED;
X    }
X}
X
X/*
X * ck_aref() - check for an aliased reference
X */
Xstatic bool
Xck_aref(d, len)
Xregister long    d;
Xregister int len;
X{
X    register int  i;
X
X    for (i=0; i < lcnt ;i++) {
X         if (locals[i].offset == d && locals[i].size == len)
X              continue;
X
X         if (overlaps(d, len, locals[i].offset, locals[i].size)) {
X              locals[i].flags |= ALIASED;
X              return TRUE;
X         }
X    }
X    return FALSE;
X}
X
Xstatic bool
Xoverlaps(d1, s1, d2, s2)
Xregister long    d1, d2;
Xint     s1, s2;
X{
X    register long e1, e2;
X
X    e1 = d1 + s1 - 1;
X    e2 = d2 + s2 - 1;
X
X    if (d1 >= d2 && d1 <= e2)   /* d1 inside d2 <=> e2 */
X         return TRUE;
X
X    if (e1 >= d2 && e1 <= e2)   /* e1 inside d2 <=> e2 */
X         return TRUE;
X
X    return FALSE;
```

```
X}
X
Xstatic void
Xsort_table()
X{
X    register int  i;
X
X    /*
X     * Remove uninteresting references from consideration:
X     *
X     * 1. Variables whose address was taken, or are aliased with another.
X     * 2. Variables that don't fit in a register.
X     */
X    for (i=0; i < lcnt ;i++) {
X         if (locals[i].flags&(ADDR_TAKEN|ALIASED) || locals[i].size > 4)
X              locals[i].ref = -locals[i].ref;
X    }
X
X    /*
X     * If paddr is set, remove any parameters from consideration. We
X     * have to do this so that things like printf (that take the address
X     * of a parameter and increment it) don't break. Only if no parameter
X     * addresses are taken, can we consider registerizing any of them.
X     */
X    if (paddr) {
X         for (i=0; i < lcnt ;i++) {
X              if (IS_PARM(i) && (locals[i].ref > 0))
X                   locals[i].ref = -locals[i].ref;
X         }
X    }
X
X    do_sort();
X}
X
Xstatic void
Xdo_sort()
X{
X    register int  i;
X    struct   linfo    l;
X
X    /*
X     * simple bubble sort
X     */
X    for (i=0; i < (lcnt-1) ;) {
X         if (locals[i].ref < locals[i+1].ref) {
X              l = locals[i];
X              locals[i] = locals[i+1];
X              locals[i+1] = l;
X              if (i > 0)
X                   i--;
X         } else
X              i++;
X    }
```

```
X}
X
X/*
X * lrewrite() - rewrite the function based on the new register assignments
X *
X * Fixing the references is easy, but we have to fix up (or add) the movem
X * instructions as well. Also, we call addinits() to initialize any registers
X * that will contain parameters.
X */
Xstatic void
Xlrewrite(bp)
XBLOCK   *bp;
X{
X    void fixref(), fixmove(), addmovem();
X    INST *findlnk();
X    register int  i;
X    register BLOCK    *cb;
X    register INST *ci;
X
X    /*
X     * First, rewrite all the references to the locals that
X     * we've reassigned to registers.
X     */
X    for (cb = bp; cb != NULL ;cb = cb->next) {
X        for (ci = cb->first; ci != NULL ;ci = ci->next) {
X            fixref(&ci->src);
X            fixref(&ci->dst);
X        }
X    }
X
X    /*
X     * If the movem's are there, just find them and fix up the
X     * register specs.
X     */
X    ci = bp->first->next;
X    if (ci != NULL && ci->opcode == MOVEM) {
X
X        /*
X         * First, add the initialization instructions.
X         */
X        addinits(bp, bp->first->next);
X
X        fixmove(&ci->src);
X
X        for (cb = bp; cb != NULL ;cb = cb->next) {
X            if (cb->flags & B_RET) {
X                for (ci=cb->last; ci != NULL ;ci=ci->prev) {
X                    if (ci->opcode == MOVEM) {
X                        fixmove(&ci->dst);
X                        return;
X                    }
X                }
X            }
```

```
X        }
X        return;
X    }
X
X#ifdef  DEBUG
X    if (debug)
X        printf("adding movem instructions\n");
X#endif
X    /*
X     * There aren't any movem instructions, so we have to add
X     * them here. What a pain...
X     */
X    addmovem(bp, findlnk(bp), TRUE);
X    addinits(bp, findlnk(bp)->next);
X
X    for (cb = bp; cb != NULL ;cb = cb->next) {
X        if (cb->last->opcode == RTS) {
X            for (ci=cb->last; ci != NULL ;ci=ci->prev) {
X                if (ci->opcode == UNLK) {
X                    addmovem(cb, ci, FALSE);
X                    return;
X                }
X            }
X        }
X    }
X    /*
X     * Reaching this point would be an error, you'd think. It means
X     * we didn't find the exit from this function. Strangely enough,
X     * this can actually happen in routines with infinite loops.
X     * Since the "return" block isn't reachable, branch optimization
X     * actually removes it. So we can't consider it an error here if
X     * we don't find any "unlk" instruction.
X     */
X}
X
Xstatic void
Xfixmove(op)
Xstruct  opnd *op;
X{
X    char *masktos();
X
X    freeop(op);
X    op->amode = ABS;
X    op->astr = strsave(masktos(nmask));
X}
X
X/*
X * findlnk() - find the LINK instruction in the given block
X *
X * When profiling, the LINK isn't the first instruction in the entry
X * block. This function lets us handle both cases cleanly.
X */
Xstatic  INST *
```

```
Xfindlnk(bp)
XBLOCK   *bp;
X{
X    INST *ip;
X
X    for (ip=bp->first; ip != NULL ;ip = ip->next) {
X         if (ip->opcode == LINK)
X              return ip;
X    }
X    return NULL;
X}
X
Xstatic void
Xaddmovem(bp, ip, is_entry)
XBLOCK   *bp;          /* block where we're working */
XINST    *ip;          /* instruction before or after the movem */
Xbool    is_entry;     /* true if we're doing the entry code */
X{
X    char *masktos();
X    register INST *ni;
X    struct   opnd *op;
X    register int  i;
X
X    if (ip == NULL)        /* no LINK found */
X         return;
X
X    /*
X     * Allocate and initialize a new instruction
X     */
X    ni = (INST *) alloc(sizeof(INST));
X
X    ni->flags = LENL;
X    ni->opcode = MOVEM;
X    ni->live = 0;
X    ni->rref = ni->rset = 0;
X
X    ni->src.areg = ni->dst.areg = 0;
X    ni->src.ireg = ni->dst.ireg = 0;
X    ni->src.disp = ni->dst.disp = 0;
X
X    /*
X     * Set up the SP reference
X     */
X    op = (is_entry) ? &ni->dst : &ni->src;
X    op->amode = (is_entry) ? REGI|DEC : REGI|INC;
X    op->areg = SP;
X
X    /*
X     * Set up the register spec operand
X     */
X    op = (is_entry) ? &ni->src : &ni->dst;
X
X    op->amode = ABS;
```

```
X    op->astr = strsave(masktos(nmask));
X
X    /*
X     * If there's only one register being used, we should really
X     * change the operand to be register direct. This way, the
X     * peephole optimization will turn the "movem" into a simple
X     * "move". Since we're adding an instruction here, we really
X     * need to make it as painless as possible.
X     */
X    if (rcnt == 1) {
X         free(op->astr);
X         op->amode = REG;
X
X         for (i=D0; i <= D7 ;i++) {
X              if (nmask & RM(i)) {
X                   op->areg = i;
X                   break;
X              }
X         }
X    }
X
X    /*
X     * Link the instruction into the block
X     */
X    if (is_entry) {
X         ni->next = ip->next;    /* link the MOVEM to its neighbors */
X         ni->prev = ip;
X
X         ip->next = ni;          /* link its neighbors to the MOVEM */
X
X         if (bp->last == ip)
X              bp->last = ni;
X         else
X              ni->next->prev = ni;
X    } else {
X         ni->next = ip;          /* link the MOVEM to its neighbors */
X         ni->prev = ip->prev;
X
X         ip->prev = ni;          /* link its neighbors to the MOVEM */
X
X         if (bp->first == ip)
X              bp->first = ni;
X         else
X              ni->prev->next = ni;
X    }
X}
X
Xstatic void
Xaddinits(bp, ip)
XBLOCK    *bp;          /* block where we're working */
XINST     *ip;          /* instruction before the moves */
X{
X    char *masktos();
```

```
X    register INST *ni;
X    struct   opnd *op;
X    register int  i;
X
X    if (ip == NULL)        /* no LINK found */
X         return;
X
X    for (i=0; i < rcnt ;i++) {
X         /*
X          * If it's a local variable, we don't have to do anything.
X          */
X         if (IS_LOCAL(i))
X              continue;
X
X         /*
X          * Allocate and initialize a new instruction
X          */
X         ni = (INST *) alloc(sizeof(INST));
X
X         switch (locals[i].size) {
X         case 1:
X              ni->flags = LENB;
X              break;
X         case 2:
X              ni->flags = LENW;
X              break;
X         case 4:
X              ni->flags = LENL;
X              break;
X         default:
X              fprintf(stderr, "Invalid length\n");
X              exit(1);
X         }
X
X         ni->opcode = MOVE;
X         ni->live = 0;
X         ni->rref = ni->rset = 0;
X         ni->src.ireg = ni->dst.ireg = 0;
X
X         /*
X          * Set up the variable reference.
X          */
X         ni->src.amode = REGID;
X         ni->src.areg  = A6;
X         ni->src.disp  = locals[i].offset;
X
X         /*
X          * Set up the register spec operand
X          */
X         ni->dst.amode = REG;
X         ni->dst.areg  = locals[i].reg;
X         ni->dst.disp  = 0;
X
```

```
X         /*
X          * Link the instruction into the block
X          */
X         ni->next = ip->next;    /* link MOVE to its neighbors */
X         ni->prev = ip;
X
X         ip->next = ni;          /* link neighbors to the MOVE */
X
X         if (bp->last == ip)
X              bp->last = ni;
X         else
X              ni->next->prev = ni;
X    }
X}
X
Xstatic void
Xfixref(op)
Xstruct  opnd *op;
X{
X    register int  i;
X
X    if (op->amode != REGID || op->areg != A6)
X         return;
X
X    /*
X     * Does the reference need to be changed?
X     */
X    for (i=0; i < rcnt ;i++) {
X         if (locals[i].offset == op->disp) {
X              op->amode = REG;
X              op->areg = locals[i].reg;
X              return;
X         }
X    }
X}
X
X/*
X * stomask() - convert a register list to a mask
X *
X * Convert a string like "Rm-Rn/Ro-Rp" or "Rm-Rn" to the appropriate
X * mask value.
X */
Xstatic int
Xstomask(s)
Xchar    *s;
X{
X    register int  mask;
X    register char *p;
X
X    mask = dorspec(s);
X
X    for (p=s; *p && *p != '/' ;p++)
X         ;
```

```
X
X    if (*p == '/')
X         mask |= dorspec(p+1);
X
X    return mask;
X}
X
X/*
X * dorspec() - convert a partial register spec
X *
X * Convert a string like "Rm" or "Rm-Rn" to a mask.
X */
Xstatic int
Xdorspec(s)
Xregister char     *s;
X{
X    register int  base;
X    register int  m, n;
X    register int  mask;
X
X    base = (s[0] == 'd') ? D0 : A0;
X
X    m = s[1] - '0' + base;
X
X    if (s[2] != '-')
X         return RM(m);
X
X    n = s[4] - '0' + base;
X
X    for (mask=0; m <= n ;m++)
X         mask |= RM(m);
X
X    return mask;
X}
X
X/*
X * masktos() - convert a register mask to a descriptive string
X *
X * Generates a string of the form "Rm/Rn/Ro/..."
X */
Xstatic char *
Xmasktos(mask)
Xregister int mask;
X{
X    static   char buf[64];
X    register char *p = buf;
X    register int  r;
X
X    for (r = D0; r <= D7 ;r++) {
X         if (mask & RM(r)) {
X              if (p != buf)
X                   *p++ = '/';
X              *p++ = 'd';
```

```
X             *p++ = (r - D0) + '0';
X        }
X    }
X    for (r = A0; r <= A7 ;r++) {
X        if (mask & RM(r)) {
X             if (p != buf)
X                  *p++ = '/';
X             *p++ = 'a';
X             *p++ = (r - A0) + '0';
X        }
X    }
X    *p = '\0';
X
X    return buf;
X}
X
X#ifdef  DEBUG
Xdump_table()
X{
X    register int  i;
X
X    printf("%d local variables and parameters found\n", lcnt);
X    for (i=0; i < lcnt ;i++) {
X        printf("len = %d\n", locals[i].size);
X        printf("%02d: disp=%3ld, len=%d ref=%2d reg=%s",
X             i, locals[i].offset,
X             locals[i].size,
X             locals[i].ref,
X             locals[i].reg >= 0 ? masktos(RM(locals[i].reg)) : "-");
X        if (locals[i].flags & ADDR_TAKEN)
X             printf(" ADDR_TAKEN");
X        if (locals[i].flags & ALIASED)
X             printf(" ALIASED");
X        printf("\n");
X    }
X}
X#endif
END_OF_FILE
if test 17767 -ne `wc -c <'top/REG.C'`; then
    echo shar: \"'top/REG.C'\" unpacked with wrong size!
fi
# end of 'top/REG.C'
fi
echo shar: End of archive 8 \(of 9\).
cp /dev/null ark8isdone
MISSING=""
for I in 1 2 3 4 5 6 7 8 9 ; do
    if test ! -f ark${I}isdone ; then
     MISSING="${MISSING} ${I}"
    fi
done
if test "${MISSING}" = "" ; then
    echo You have unpacked all 9 archives.
```

```
    rm -f ark[1-9]isdone ark[1-9][0-9]isdone
else
    echo You still need to unpack the following archives:
    echo "        " ${MISSING}
fi
##  End of shell archive.
exit 0


From 7176019994171419317
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,881f4913a4e86871,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-25 10:15:58 PST
Path: utzoo!attcan!utgpu!jarvis.csri.toronto.edu!mailrus!cs.utexas.edu!sun-barr!
newstop!sun!sunfedcomm!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v02i100: sozobon1.2 -- Update to Sozobon C compiler  part09/09
Keywords: shar, C
Message-ID: <34613@grapevine.uucp>
Date: 25 Oct 89 17:15:58 GMT
Sender: daemon@sunfedcomm
Reply-To: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Lines: 871
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: ncar.ucar.edu!dunike!onecom!wldrdg!hans (Johann Ruegg)
Posting-number: Volume 2, Issue 100
Archive-name: sozobon1.2/part09

#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#        "End of archive 9 (of 9)."
# Contents:  top/PEEP2.C
# Wrapped by koreth@panarthea on Tue Oct 24 18:40:47 1989
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'top/PEEP2.C' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'top/PEEP2.C'\"
else
echo shar: Extracting \"'top/PEEP2.C'\" \(17904 characters\)
sed "s/^X//" >'top/PEEP2.C' <<'END_OF_FILE'
X/* Copyright (c) 1988 by Sozobon, Limited.  Author: Tony Andrews
X *
X * Permission is granted to anyone to use this software for any purpose
X * on any computer system, and to redistribute it freely, with the
X * following restrictions:
X * 1) No charge may be made other than reasonable charges for reproduction.
X * 2) Modified versions must be clearly marked as such.
```

```
X * 3) The authors are not responsible for any harmful consequences
X *    of using this software, even if they result from defects in it.
X */
X
X/*
X * 2-instruction peephole optimizations
X */
X
X#include "top.h"
X
X
X/*
X * ipeep2(bp, i1) - look for 2-instruction optimizations at the given inst.
X */
Xstatic  bool
Xipeep2(bp, i1)
XBLOCK   *bp;
Xregister INST    *i1;
X{
X    register INST *i2;     /* the next instruction */
X    register INST *ti2;         /* "temporary" next inst */
X
X    register int  op1, op2;/* opcodes, for speed */
X    register int  sm1, dm1;/* src, dst amode inst. 1 */
X    register int  dr1;     /* dest. reg. inst. 1 */
X
X    i2  = i1->next;
X    op1 = i1->opcode;
X    op2 = i2->opcode;
X
X    sm1 = i1->src.amode;
X    dm1 = i1->dst.amode;
X    dr1 = i1->dst.areg;
X
X    /*
X     * Avoid stack fix-ups after a call if possible.
X     */
X
X    /*
X     *   addq #4,sp
X     *   ... stuff that doesn't use SP ...
X     *   move.l   ?,-(sp)       =>   move.l   ?,(sp)
X     */
X    if (op1 == ADDQ && sm1 == IMM && i1->src.disp == 4 &&
X        dm1 == REG && dr1 == SP) {
X
X        ti2 = i2;
X        while (!uses(ti2, SP)) {
X             if (ti2->next == NULL)
X                  goto end2;
X             ti2 = ti2->next;
X        }
X
```

```
X        if (ti2->opcode == MOVE && ti2->flags == LENL &&
X            ti2->dst.amode == (REGI|DEC) && ti2->dst.areg == SP) {
X             ti2->dst.amode = REGI;
X             delinst(bp, i1);
X             DBG(printf("%d ", __LINE__))
X             return TRUE;
X        }
X    }
Xend2:
X
X    /*
X     *   addq #2,sp
X     *   ... stuff that doesn't use SP ...
X     *   move.w   ?,-(sp)       =>   move.w   ?,(sp)
X     */
X    if (op1 == ADDQ && sm1 == IMM && i1->src.disp == 2 &&
X        dm1 == REG && dr1 == SP) {
X
X        ti2 = i2;
X        while (!uses(ti2, SP)) {
X             if (ti2->next == NULL)
X                  goto end3;
X             ti2 = ti2->next;
X        }
X
X        if (ti2->opcode == MOVE && ti2->flags == LENW &&
X            ti2->dst.amode == (REGI|DEC) && ti2->dst.areg == SP) {
X             ti2->dst.amode = REGI;
X             delinst(bp, i1);
X             DBG(printf("%d ", __LINE__))
X             return TRUE;
X        }
X    }
Xend3:
X
X    /*
X     * Avoid "tst" instructions following instructions that
X     * set the Z flag.
X     */
X
X    /*
X     *   move.x   X, Y     =>   move.x   X, Y
X     *   tst.x    X or Y            ...deleted...
X     *   beq/bne                 beq/bne
X     *
X     *   Where Y is not An, because "movea" doesn't set the
X     *   zero flag.
X     */
X    if (bp->last == i2 && (bp->bcode == BEQ || bp->bcode == BNE) &&
X        op1 == MOVE && op2 == TST &&
X        i1->flags == i2->flags) {
X
X        /*
```

```
X         * If pre-decrement is set on the dest. of the move,
X         * don't let that screw up the operand comparison.
X         */
X        if (dm1 & DEC)
X            dm1 &= ~DEC;
X
X        if (opeq(&i1->dst, &i2->src) || opeq(&i1->src, &i2->src)) {
X            if (dm1 != REG || ISD(dr1)) {
X                delinst(bp, i2);
X                DBG(printf("%d ", __LINE__))
X                return TRUE;
X            }
X        }
X    }
X
X    /*
X     *   and.x    X, Y     =>   and.x    X, Y
X     *   tst.x    X or Y            ...deleted...
X     *   beq/bne                beq/bne
X     *
X     *   Where Y is not An, because "movea" doesn't set the
X     *   zero flag.
X     */
X    if (bp->last == i2 && (bp->bcode == BEQ || bp->bcode == BNE) &&
X        op1 == AND && op2 == TST &&
X        i1->flags == i2->flags) {
X
X        /*
X         * If pre-decrement is set on the dest. of the move,
X         * don't let that screw up the operand comparison.
X         */
X        if (dm1 & DEC)
X            dm1 &= ~DEC;
X
X        if (opeq(&i1->dst, &i2->src) || opeq(&i1->src, &i2->src)) {
X            if (dm1 != REG || ISD(dr1)) {
X                delinst(bp, i2);
X                DBG(printf("%d ", __LINE__))
X                return TRUE;
X            }
X        }
X    }
X
X    /*
X     *   ext.x    Dn       =>   ext.x    Dn
X     *   tst.x    Dn            ...deleted...
X     *   beq/bne                beq/bne
X     *
X     *   Where Y is not An, because "movea" doesn't set the
X     *   zero flag.
X     */
X    if (bp->last == i2 && (bp->bcode == BEQ || bp->bcode == BNE) &&
X        op1 == EXT && op2 == TST &&
```

```
X        i1->flags == i2->flags) {
X
X         if (sm1 == REG && ISD(i1->src.areg) &&
X             i2->src.amode == REG && i1->src.areg == i2->src.areg) {
X              delinst(bp, i2);
X              DBG(printf("%d ", __LINE__))
X              return TRUE;
X         }
X    }
X
X    /*
X     *   move.?    X, Dn         =>   move.?    X, Dn
X     *   ext.?     Dn            ...deleted...
X     *   beq/bne                 beq/bne
X     *
X     *   Where Dn is dead after the "ext".
X     */
X    if (bp->last == i2 && (bp->bcode == BEQ || bp->bcode == BNE) &&
X        op1 == MOVE && op2 == EXT) {
X
X         if (dm1 == REG && ISD(dr1) &&
X             i2->src.amode == REG && dr1 == i2->src.areg) {
X              if ((i2->live & RM(i2->src.areg)) == 0) {
X                   delinst(bp, i2);
X                   DBG(printf("%d ", __LINE__))
X                   return TRUE;
X              }
X         }
X    }
X
X    /*
X     *   ext.l     Dm        =>   ...deleted...
X     *   tst.l     Dm             tst.w     Dm
X     *
X     *   where Dm is dead after the "tst".
X     */
X    if (op1 == EXT && op2 == TST &&
X        ((i1->flags & LENL) != 0) &&  ((i2->flags & LENL) != 0) &&
X        (i1->src.areg == i2->src.areg) && ISD(i1->src.areg)) {
X
X         if ((i2->live & RM(i2->src.areg)) == 0) {
X              i2->flags = LENW;
X              delinst(bp, i1);
X              DBG(printf("%d ", __LINE__))
X              return TRUE;
X         }
X    }
X
X    /*
X     *   ext.l     Dm        =>   ...deleted...
X     *   ???  N(An,Dm.l), ??         ???  N(An,Dm.w), ??
X     *
X     *   Where Dm is dead
```

```
X     */
X    if ((op1 == EXT) && (i1->flags & LENL) &&
X        (i2->src.amode == (REGIDX|XLONG)) &&
X        (i1->src.areg == i2->src.ireg)) {
X
X         if ((i2->live & RM(i1->src.areg)) == 0) {
X              i2->src.amode &= ~XLONG;
X              delinst(bp, i1);
X              DBG(printf("%d ", __LINE__))
X              return TRUE;
X         }
X    }
X
X    /*
X     *   ext.l     Dm        =>   ...deleted...
X     *   ???  ??, N(An,Dm.l)          ???  ??, N(An,Dm.w)
X     *
X     *   Where Dm is dead
X     */
X    if ((op1 == EXT) && (i1->flags & LENL) &&
X        (i2->dst.amode == (REGIDX|XLONG)) &&
X        (i1->src.areg == i2->dst.ireg)) {
X
X         if ((i2->live & RM(i1->src.areg)) == 0) {
X              i2->dst.amode &= ~XLONG;
X              delinst(bp, i1);
X              DBG(printf("%d ", __LINE__))
X              return TRUE;
X         }
X    }
X
X    /*
X     * Avoid intermediate registers.
X     */
X
X    /*
X     *   move.x    X, Dm         =>   INST.x    X, Dn
X     *   INST.x    Dm, Dn
X     *
X     *   where Dm is dead, and INST is one of: add, sub, and, or, cmp
X     */
X    if ((op1 == MOVE) &&
X        ((op2==ADD)||(op2==SUB)||(op2==AND)||(op2==OR)||(op2==CMP)) &&
X        (i1->flags == i2->flags) &&
X        (dm1 == REG) && ISD(dr1) &&
X        (i2->src.amode == REG) && ISD(i2->src.areg) &&
X        (dr1 == i2->src.areg) &&
X        (i2->dst.amode == REG) && ISD(i2->dst.areg)) {
X
X         if ((i2->live & RM(i2->src.areg)) == 0) {
X
X              i1->opcode = i2->opcode;
X              i1->dst.areg = i2->dst.areg;
```

```
X
X            delinst(bp, i2);
X            DBG(printf("%d ", __LINE__))
X            return TRUE;
X        }
X    }
X
X    /*
X     * Silly moves
X     */
X
X    /*
X     *   move.x    X, Y      =>   move.x    X, Y
X     *   move.x    Y, X
X     */
X    if ((op1 == MOVE) && (op2 == MOVE) &&
X        (i1->flags == i2->flags) &&
X        opeq(&i1->src, &i2->dst) && opeq(&i1->dst, &i2->src) &&
X        ((i1->src.amode & (INC|DEC)) == 0) &&
X        ((i1->dst.amode & (INC|DEC)) == 0)) {
X
X            delinst(bp, i2);
X        DBG(printf("%d ", __LINE__))
X            return TRUE;
X    }
X
X    /*
X     *   move.x    X, Y      =>   move.x    X, Rn
X     *   move.x    Y, Rn             move.x    Rn, Y
X     *
X     *   where Y isn't INC or DEC, and isn't register direct
X     */
X    if ((op1 == MOVE) && (op2 == MOVE) && (i2->dst.amode == REG) &&
X        opeq(&i1->dst, &i2->src) && ((i1->dst.amode & (INC|DEC)) == 0) &&
X        (i1->flags == i2->flags) && (i1->dst.amode != REG)) {
X
X        freeop(&i1->dst);
X        i1->dst = i2->dst;
X        i2->dst = i2->src;
X        i2->src = i1->dst;
X
X        DBG(printf("%d ", __LINE__))
X        return TRUE;
X    }
X
X    /*
X     *   move.x    Dm, X         =>   move.x    Dm, X
X     *   move.x    X, Y          move.x    Dm, Y
X     *
X     * Where 'x' is the same, and 'X' has no side-effects.
X     */
X    if ((op1 == MOVE) && (op2 == MOVE) &&
X        (sm1 == REG) && ISD(i1->src.areg) &&
```

```
X        (i1->flags == i2->flags) && opeq(&i1->dst, &i2->src) &&
X        ((dm1 & (DEC|INC)) == 0)) {
X
X
X        freeop(&i2->src);
X        i2->src = i1->src;
X        DBG(printf("%d ", __LINE__))
X             return TRUE;
X    }
X
X    /*
X     *   move.?   Rm, Rn        =>   move.?   Rm, Rn
X     *   ... stuff ...          ... stuff ...
X     *   move.?   Rm, Rn
X     *
X     *   where "stuff" doesn't set Rm or Rn. Also make sure that
X     *   the second move isn't followed by a conditional branch.
X     *   In that case leave everything alone since the branch
X     *   probably relies on flags set by the move.
X     */
X    if ((op1 == MOVE) && (sm1 == REG) && (dm1 == REG)) {
X        int  s1 = i1->src.areg; /* source reg of inst. 1 */
X
X        ti2 = i2;
X        while (ti2 != NULL && !sets(ti2, s1) && !sets(ti2, dr1)) {
X
X             if ((ti2->opcode==MOVE) && (i1->flags==ti2->flags) &&
X                 (ti2->src.amode==REG) && (ti2->dst.amode==REG) &&
X                 (i1->src.areg == ti2->src.areg) &&
X                 (i1->dst.areg == ti2->dst.areg) &&
X                 ((bp->last != ti2) || (bp->bcond == NULL)) ) {
X
X                      delinst(bp, ti2);
X                  DBG(printf("%d ", __LINE__))
X                      return TRUE;
X             }
X             ti2 = ti2->next;
X        }
X    }
X
X    /*
X     *   move.l   Am, Dn        =>   move.l   Am, Ao
X     *   ... stuff ...          ... stuff ...
X     *   move.l   Dn, Ao
X     *
X     *   where "stuff" doesn't set Dn.
X     */
X    if ((op1 == MOVE) && (i1->flags == LENL) &&
X        (sm1 == REG) && ISA(i1->src.areg) &&
X        (dm1 == REG) && ISD(dr1)) {
X
X        ti2 = i2;
X        while (!sets(ti2, dr1)) {
```

```
X
X            if ((ti2->opcode == MOVE) && (ti2->flags == LENL) &&
X                (ti2->src.amode == REG) && ISD(ti2->src.areg) &&
X                (ti2->dst.amode == REG) && ISA(ti2->dst.areg) &&
X                (dr1 == ti2->src.areg)) {
X
X                /*
X                 * If the intermediate register isn't dead,
X                 * then we have to keep using it.
X                 */
X                if ((ti2->live & RM(ti2->src.areg)) != 0)
X                    goto end14;
X
X                i1->dst.areg = ti2->dst.areg;
X
X                    delinst(bp, ti2);
X                DBG(printf("%d ", __LINE__))
X                    return TRUE;
X            }
X
X            if (ti2->next == NULL)
X                goto end14;
X
X            ti2 = ti2->next;
X        }
X    }
Xend14:
X
X    /*
X     *   move.l    Dm, An        =>   move.l    Dm, Ao
X     *   lea  (An), Ao
X     *
X     *   where An is dead
X     */
X    if ((op1 == MOVE) && (op2 == LEA) &&
X        (sm1 == REG) && ISD(i1->src.areg) &&
X        (dm1 == REG) && ISA(dr1) &&
X        (i2->src.amode == REGI) && (i2->dst.amode == REG) &&
X        ISA(i2->dst.areg) && (dr1 == i2->src.areg)) {
X
X        if ((i2->live & RM(i2->src.areg)) == 0) {
X
X            i1->dst.areg = i2->dst.areg;
X
X            delinst(bp, i2);
X            DBG(printf("%d ", __LINE__))
X            return TRUE;
X        }
X    }
X
X    /*
X     *   lea  X, An         =>   lea  X, Ao
X     *   lea  (An), Ao
```

```
X     *
X     *   where An is dead
X     */
X    if ((op1 == LEA) && (op2 == LEA) &&
X        (i2->src.amode == REGI) && (dr1 == i2->src.areg)) {
X
X        if ((i2->live & RM(i2->src.areg)) == 0) {
X
X             i1->dst.areg = i2->dst.areg;
X
X             delinst(bp, i2);
X             DBG(printf("%d ", __LINE__))
X             return TRUE;
X        }
X    }
X
X    /*
X     *   lea  N(Am), Am=>
X     *   ?    (Am)[,...]         ?    N(Am)[,...]
X     *
X     *   Where Am is either dead after the second instruction or
X     *   is a direct destination of the second instruction.
X     */
X    if ((op1 == LEA) && (i1->src.amode == REGID) &&
X        (i1->src.areg == i1->dst.areg) &&
X        (i2->src.amode == REGI) && (i1->dst.areg == i2->src.areg)) {
X
X        if (((i2->live & RM(i2->src.areg)) == 0) ||
X           ((i2->dst.amode == REG) && (i2->dst.areg == i2->src.areg))) {
X             i2->src.amode = REGID;
X             i2->src.disp = i1->src.disp;
X             delinst(bp, i1);
X             DBG(printf("%d ", __LINE__))
X             return TRUE;
X        }
X    }
X
X    /*
X     *   lea  N(Am), Am=>
X     *   ?    X, (Am)            ?    X, N(Am)
X     *
X     *   Where X doesn't reference Am, and Am is dead after the
X     *   second instruction.
X     */
X    if ((op1 == LEA) && (i1->src.amode == REGID) &&
X        (i1->src.areg == i1->dst.areg) &&
X        (i2->dst.amode == REGI) && (i1->dst.areg == i2->dst.areg)) {
X
X        if (((i2->live & RM(i2->dst.areg)) == 0) &&
X            ((i2->src.amode == IMM) || (i2->src.amode == ABS) ||
X             (i2->src.areg != i2->dst.areg))) {
X             i2->dst.amode = REGID;
X             i2->dst.disp = i1->src.disp;
```

```
X             delinst(bp, i1);
X             DBG(printf("%d ", __LINE__))
X             return TRUE;
X         }
X     }
X
X     /*
X      *   lea  X, Am         =>   ...deleted...
X      *   clr.x    (Am)          clr.x    X
X      *
X      *   where Am is dead
X      */
X     if ((op1 == LEA) && (op2 == CLR) && i2->src.amode == REGI &&
X         (i1->dst.areg == i2->src.areg)) {
X
X         if ((i2->live & RM(i2->src.areg)) == 0) {
X             i2->src = i1->src;
X
X             delinst(bp, i1);
X             DBG(printf("%d ", __LINE__))
X             return TRUE;
X         }
X     }
X
X     /*
X      *   lea  X, Am         =>   ...deleted...
X      *   move.x   Y, (Am)           move.x   Y, X
X      *
X      *   where Am is dead
X      */
X     if ((op1 == LEA) && (op2 == MOVE) && i2->dst.amode == REGI &&
X         (i1->dst.areg == i2->dst.areg)) {
X
X         if ((i2->live & RM(i2->dst.areg)) == 0) {
X             i2->dst = i1->src;
X
X             delinst(bp, i1);
X             DBG(printf("%d ", __LINE__))
X             return TRUE;
X         }
X     }
X
X     /*
X      *   lea  X, Am         =>   ...deleted...
X      *   move.x   (Am), Y           move.x   X, Y
X      *
X      *   where Am is dead
X      */
X     if ((op1 == LEA) && (op2 == MOVE) && i2->src.amode == REGI &&
X         (i1->dst.areg == i2->src.areg)) {
X
X         if ((i2->live & RM(i2->src.areg)) == 0) {
X             i2->src = i1->src;
```

```
X
X            delinst(bp, i1);
X            DBG(printf("%d ", __LINE__))
X            return TRUE;
X        }
X    }
X
X    /*
X     *   move.x    Dm, X         =>   move.x    Dm, X
X     *   cmp.x     #N, X              cmp.x     #N, Dm
X     *
X     *   Where X isn't register direct.
X     *
X     *   Since X generally references memory, we can compare
X     *   with the register faster.
X     */
X    if ((op1 == MOVE) && (op2 == CMP) &&
X        (i1->flags == i2->flags) && (i2->src.amode == IMM) &&
X        (sm1 == REG) && ISD(i1->src.areg) && (dm1 != REG) &&
X        opeq(&i1->dst, &i2->dst) && ((dm1 & (INC|DEC)) == 0)) {
X
X        freeop(&i2->dst);
X        i2->dst.amode = REG;
X        i2->dst.areg = i1->src.areg;
X
X        DBG(printf("%d ", __LINE__))
X        return TRUE;
X    }
X
X
X    /*
X     * Try to use register indirect w/ displacement and/or index
X     */
X
X    /*
X     *   add.l     Am, Dn        =>   lea  0(Am,Dn.l), Ao
X     *   move.l    Dn, Ao
X     *
X     *   where Dn is dead
X     */
X    if ((op1 == ADD) && (op2 == MOVE) &&
X        (sm1 == REG) && ISA(i1->src.areg) &&
X        (dm1 == REG) && ISD(dr1) &&
X        (i2->src.amode == REG) && ISD(i2->src.areg) &&
X        (i2->dst.amode == REG) && ISA(i2->dst.areg) &&
X        (dr1 == i2->src.areg) &&
X        (i1->flags & LENL) && (i2->flags & LENL)) {
X
X        if ((i2->live & RM(i2->src.areg)) == 0) {
X
X            i2->opcode = LEA;
X            i2->flags = 0;
X
```

```
X             i2->src.amode = REGIDX|XLONG;
X             i2->src.disp = 0;
X             i2->src.areg = i1->src.areg;
X             i2->src.ireg = dr1;
X
X             delinst(bp, i1);
X             DBG(printf("%d ", __LINE__))
X             return TRUE;
X         }
X    }
X
X    /*
X     *   add.l    Dm, An        =>   move.x   0(An,Dm.l), Do
X     *   move.x   (An), Do
X     *
X     *   where An is dead
X     */
X    if ((op1 == ADD) && (op2 == MOVE) &&
X        (sm1 == REG) && ISD(i1->src.areg) &&
X        (dm1 == REG) && ISA(dr1) &&
X        (i2->src.amode == REGI)&& ISA(i2->src.areg) &&
X        (i2->dst.amode == REG) && ISD(i2->dst.areg) &&
X        (dr1 == i2->src.areg) && (i1->flags & LENL)) {
X
X         if ((i2->live & RM(i2->src.areg)) == 0) {
X
X             i2->src.amode = REGIDX|XLONG;
X             i2->src.disp = 0;
X             i2->src.ireg = i1->src.areg;
X
X             delinst(bp, i1);
X             DBG(printf("%d ", __LINE__))
X             return TRUE;
X         }
X    }
X
X    /*
X     *   lea  N(Am), An =>   lea  N(Am,Do.l), An
X     *   add.l    Do, An
X     *
X     */
X    if ((op1 == LEA) && (op2 == ADD) &&
X        (sm1 == REGID) &&
X        (i2->src.amode == REG) && ISD(i2->src.areg) &&
X        (i2->dst.amode == REG) && ISA(i2->dst.areg) &&
X        (dr1 == i2->dst.areg) && D8OK(i1->src.disp)) {
X
X         i1->src.amode = REGIDX|XLONG;
X         i1->src.ireg = i2->src.areg;
X         delinst(bp, i2);
X         DBG(printf("%d ", __LINE__))
X         return TRUE;
X    }
```

```
X
X
X
X    /*
X     * Try to use the pre-decrement and post-increment modes
X     * whenever possible.
X     */
X
X    /*
X     *   sub.l    #1, Am
X     *   ... stuff ...
X     *   ???.b    ..(Am).. =>   ???.b    ..-(Am)..
X     *
X     *   Nothing in "stuff" can refer to Am.
X     */
X    if ((op1 == SUB) && (i1->flags & LENL) &&
X        (sm1 == IMM) && (i1->src.disp == 1) &&
X        (dm1 == REG) && ISA(dr1)) {
X
X        while (i2 != NULL) {
X
X            if (i2->src.amode == REGI && i2->src.areg == dr1) {
X
X                if ((i2->flags & LENB) == 0)
X                    goto end24;
X
X                i2->src.amode |= DEC;
X
X                delinst(bp, i1);
X                DBG(printf("%d ", __LINE__))
X                return TRUE;
X            }
X            if (i2->dst.amode == REGI && i2->dst.areg == dr1) {
X
X                if ((i2->flags & LENB) == 0)
X                    goto end24;
X
X                i2->dst.amode |= DEC;
X
X                delinst(bp, i1);
X                DBG(printf("%d ", __LINE__))
X                return TRUE;
X            }
X
X            if (uses(i2, RM(dr1)))
X                goto end24;
X
X            if (i2->next == NULL)
X                goto end24;
X            else
X                i2 = i2->next;
X
X        }
```

```
X    }
Xend24:
X
X    /*
X     *   sub.l    #2, Am
X     *   ... stuff ...
X     *   ???.w    ..(Am).. =>   ???.w    ..-(Am)..
X     *
X     *   Nothing in "stuff" can refer to Am.
X     */
X    if ((op1 == SUB) && (i1->flags & LENL) &&
X        (sm1 == IMM) && (i1->src.disp == 2) &&
X        (dm1 == REG) && ISA(dr1)) {
X
X         while (i2 != NULL) {
X
X              if (i2->src.amode == REGI && i2->src.areg == dr1) {
X
X                   if ((i2->flags & LENW) == 0)
X                        goto end26;
X
X                   i2->src.amode |= DEC;
X
X                   delinst(bp, i1);
X                   DBG(printf("%d ", __LINE__))
X                   return TRUE;
X              }
X              if (i2->dst.amode == REGI && i2->dst.areg == dr1) {
X
X                   if ((i2->flags & LENW) == 0)
X                        goto end26;
X
X                   i2->dst.amode |= DEC;
X
X                   delinst(bp, i1);
X                   DBG(printf("%d ", __LINE__))
X                   return TRUE;
X              }
X
X              if (uses(i2, RM(dr1)))
X                   goto end26;
X
X              if (i2->next == NULL)
X                   goto end26;
X              else
X                   i2 = i2->next;
X
X         }
X    }
Xend26:
X
X    /*
X     *   sub.l    #4, Am
```

```
X     *   ... stuff ...
X     *   ???.l     ..(Am).. =>   ???.l     ..-(Am)..
X     *
X     *   Nothing in "stuff" can refer to Am.
X     */
X    if ((op1 == SUB) && (i1->flags & LENL) &&
X        (sm1 == IMM) && (i1->src.disp == 4) &&
X        (dm1 == REG) && ISA(dr1)) {
X
X         while (i2 != NULL) {
X
X              if (i2->src.amode == REGI && i2->src.areg == dr1) {
X
X                   if ((i2->flags & LENL) == 0)
X                        goto end28;
X
X                   i2->src.amode |= DEC;
X
X                   delinst(bp, i1);
X                   DBG(printf("%d ", __LINE__))
X                   return TRUE;
X              }
X              if (i2->dst.amode == REGI && i2->dst.areg == dr1) {
X
X                   if ((i2->flags & LENL) == 0)
X                        goto end28;
X
X                   i2->dst.amode |= DEC;
X
X                   delinst(bp, i1);
X                   DBG(printf("%d ", __LINE__))
X                   return TRUE;
X              }
X
X              if (uses(i2, RM(dr1)))
X                   goto end28;
X
X              if (i2->next == NULL)
X                   goto end28;
X              else
X                   i2 = i2->next;
X
X         }
X    }
Xend28:
X
X    return FALSE;
X}
X
X/*
X * peep2(bp) - scan blocks starting at 'bp'
X */
Xbool
```

```
Xpeep2(bp)
Xregister BLOCK    *bp;
X{
X    register INST *ip;
X    register bool changed = FALSE;
X
X    DBG(printf("p2 :"))
X
X    for (; bp != NULL ;bp = bp->next) {
X         for (ip = bp->first; ip != NULL && ip->next != NULL ;) {
X              if (ipeep2(bp, ip)) {
X                   changed = TRUE;
X                   s_peep2++;
X                   bprep(bp);
X                   /*
X                    * If we had a match, then either instruction
X                    * may have been deleted, so the safe thing to
X                    * do is to go to the next block.
X                    */
X                   break;
X              } else
X                   ip = ip->next;
X         }
X    }
X    DBG(printf("\n"); fflush(stdout))
X    return changed;
X}
END_OF_FILE
if test 17904 -ne `wc -c <'top/PEEP2.C'`; then
    echo shar: \"'top/PEEP2.C'\" unpacked with wrong size!
fi
# end of 'top/PEEP2.C'
fi
echo shar: End of archive 9 \(of 9\).
cp /dev/null ark9isdone
MISSING=""
for I in 1 2 3 4 5 6 7 8 9 ; do
    if test ! -f ark${I}isdone ; then
     MISSING="${MISSING} ${I}"
    fi
done
if test "${MISSING}" = "" ; then
    echo You have unpacked all 9 archives.
    rm -f ark[1-9]isdone ark[1-9][0-9]isdone
else
    echo You still need to unpack the following archives:
    echo "        " ${MISSING}
fi
##  End of shell archive.
exit 0


From 8546893278349379992
```

X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,84264ba8774c89f7,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,84264ba8774c89f7,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-31 13:10:19 PST
Xref: utzoo comp.sources.atari.st:254 comp.sys.atari.st:21031
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!wuarchive!brutus.cs.uiuc.edu!
apple!sun-barr!newstop!sun!sunfedcomm!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st,comp.sys.atari.st
Subject: v03INF2: Index of volumes 1 and 2
Message-ID: <34652@grapevine.uucp>
Date: 31 Oct 89 21:10:19 GMT
Expires: 30 Nov 89 21:10:18 GMT
Sender: daemon@sunfedcomm
Lines: 209
Approved: koreth@panarthea.ebay.sun.com (Steven Grimm)

Submitted-by: koreth@panarthea.ebay.sun.com (Steven Grimm)
Posting-number: Volume 3, Info 2
Archive-name: Index

(See the companion article for information about retrieving these files
 from an archive site.)

-- Volume 1 --

volume1/intro Introduction to comp.sources.atari.st
volume1/basic-uud Uudecode in BASIC
volume1/bootcopy Automatic file copier for RAMdisk
volume1/joyread Joystick-reading routines
volume1/cc/part01 C compiler driver (Alcyon)
volume1/cc/part02
volume1/cc/part03
volume1/cc/part04
volume1/mkmfor Something for Absoft Fortran
volume1/make/part01 Program maintainer
volume1/make/part02
volume1/make/part03
volume1/make/part04
volume1/make/part05
volume1/make/part06
volume1/make/part07
volume1/blank Fast disk eraser
volume1/mmdummy Megamax dummy routines
volume1/fdum Megamax floating-point dummy routines
volume1/idle Screen saver
volume1/autotime Time setter for auto folder
volume1/time Time setter
volume1/autodisk Copy files to RAMdisk
volume1/dvist32/part01 DVI previewer
volume1/dvist32/part02

```
volume1/dvist32/part03
volume1/dvist32/part04
volume1/dvist32/part05
volume1/dvist32/part06
volume1/dvist32/part07
volume1/dvist32/part08
volume1/dvist32/part09
volume1/cache           Disk cache
volume1/vdit            VDI interface with GDOS font support
volume1/flip/part01         Double-sided disk copier
volume1/flip/part02
volume1/rzsz/part01         UNIX end for zmdm, below
volume1/rzsz/part02
volume1/rzsz/part03
volume1/rzsz/part04
volume1/rzsz/part05
volume1/zmdm/part01         Zmodem file transfer program
volume1/zmdm/part02
volume1/zmdm/part03
volume1/zmdm/part04
volume1/zmdm/part05
volume1/zmdm/part06
volume1/zmdm/part07
volume1/zmdm/part08
volume1/ops5/part01         OPS5 production system in LISP
volume1/ops5/part02
volume1/ops5/part03
volume1/assist2/part01      Multifunction desk accessory
volume1/assist2/part02
volume1/errorchk        Verbose display of error codes
volume1/joinprnt        Echo console output to printer
volume1/color           Color-selection dialog box
volume1/rsctoc              Convert resource to C source
volume1/fastdump        Text-only screen dump
volume1/readme12        Online help system
volume1/newstart        Fancy printing
volume1/varlint
volume1/flip27/part01       Double-sided disk copier
volume1/flip27/part02
volume1/poplib              Shared libraries, with examples
volume1/uupc/part01         UUCP clone (pre-pre-release)
volume1/uupc/part02
volume1/uupc/part03
volume1/uupc/part04
volume1/gnufp           GNU floating point routines
volume1/scroll              Fast screen scroller
volume1/break           Stop programs with a keypress
volume1/newbreak        New version of above
volume1/timecode        Determine 68000 code speed
volume1/achlib/part01       Memory management and other routines for MadMac
volume1/achlib/part02
volume1/sttar           Unix-style "tar" utility
volume1/d110edit        D110 patch editor (MIDI)
```

volume1/raymovi2/part01 Raytrace movie generator
volume1/raymovi2/part02
volume1/gstlink Faster linker for GST object files
volume1/mx2v230/part01 Multitasking kernel with utilities
volume1/mx2v230/part02
volume1/mx2v230/part03
volume1/mx2v230/part04
volume1/mx2v230/part05
volume1/mx2v230/part06
volume1/mx2v230/part07
volume1/mx2v230/part08
volume1/limiter Limit uppermost hard disk device letter
volume1/d110edit Patch editor for D110 synthesizer
volume1/mt32edit Patch editor for MT32 synthesizer
volume1/starchrt/part01 Diffs to Unix starchart program for ST
volume1/starchrt/part02
volume1/passm/part01 Logic circuit assembler
volume1/passm/part02
volume1/leolife Game of Life

-- Volume 2 --

volume2/sc_src/part01 Source to Sozobon C compiler
volume2/sc_src/part02
volume2/sc_src/part03
volume2/sc_src/part04
volume2/sc_src/part05
volume2/sc_src/part06
volume2/sc_src/part07
volume2/sc_src/part08
volume2/sc_src/part09
volume2/dlibs12/part01 Source for DLibs C libraries
volume2/dlibs12/part02
volume2/dlibs12/part03
volume2/shar Pack and unpack shell archives
volume2/keyfix Fix Gulam for non-American keyboards
volume2/sed/part01 GNU sed (stream editor)
volume2/sed/part02
volume2/sort/part01 UNIX-style sort utility
volume2/sort/part02
volume2/uw/part01 Unix Windows windowing terminal software
volume2/uw/part02
volume2/uw/part03
volume2/uw/part04
volume2/uw/part05
volume2/uuduue Dumas uuencode/decode programs
volume2/mwstring dpANS string.h for MWC
volume2/mailtruk GFA BASIC uucp software
volume2/format83 Make Intel ASCII Hex files
volume2/mailtruk.11 Version 1.1 of GFA-BASIC uucp software
volume2/1stjet 1st-Word configuration file for DeskJet
volume2/1stjethv 1st-Word config file to use helvetica font on DeskJet
volume2/cpboot Copy executable boot sectors between floppy formats

```
volume2/du         List disk usage in directory trees
volume2/fplib10         Floating point routines for Sozobon C
volume2/prtset          Set printer parameters
volume2/hexify          Hexidecimal encode/decode for mailing binaries
volume2/nutshell        Use a shell from within a GEM prog
volume2/texchang        Change files for BibTeX, Tangle, and Weave
volume2/deskset         Re-read DESKTOP.INF and set up system accordingly
volume2/which      Find programs using PATH env. variable
volume2/atime89         Set GEMDOS time from keyboard clock
volume2/fplib20/part01      New floating-point library for Sozobon C
volume2/fplib20/part02
volume2/decky      Connect DEC keyboard to ST
volume2/djreset         Reset DeskJet Plus
volume2/modedit         Hardwire an editor into German Modula-2
volume2/overscan        Software driver for increased screen size
volume2/gnutar/part01       GNU's version of UNIX "tar"
volume2/gnutar/part02
volume2/gnutar/part03
volume2/gnutar/part04
volume2/gnutar/part05
volume2/gnutar/part06
volume2/gnutar/part07
volume2/gnutar/part08
volume2/coloremu        Color emulator for mono screens
volume2/gccgem/part01       GEM libraries for GCC
volume2/gccgem/part02
volume2/gccgem/part03
volume2/gccgem/part04
volume2/gccgem/part05
volume2/gccgem/part06
volume2/postmort        Analyze post-bomb information
volume2/crunch          Executable file compressor
volume2/environ         Set environment variables from AUTO folder
volume2/viruskil        Virus killer
volume2/norad      Earth satellite tracking program
volume2/diff       File difference utility
volume2/emacs.diff/part01       Diffs to GNU-EMACS 18.52 for Atari
volume2/emacs.diff/part02
volume2/protect         Virus killer/protector
volume2/ff31       File finder version 3.1
volume2/gcclib2/part01      Patches to GCC library
volume2/gcclib2/part02
volume2/gcclib2/part03
volume2/gcclib2/part04
volume2/gcclib2/part05
volume2/gcclib2/part06
volume2/gcclib2/part07
volume2/gcclib2-patch1      Patch for GCC library
volume2/mandel          Fast Mandelbrot set generator
volume2/gfadiff         File difference generator
volume2/nolock          Disable Caps-Lock key
volume2/builds09        Build Motorola S-records from .ttp files
volume2/colorca         Cellular automata in color
```

```
volume2/eternal3       Reset-proof ramdisk
volume2/degadesk       Load DEGAS .P?3 files onto desktop
volume2/truchet        Display random Truchet tilings on mono screens
volume2/yaff      Yet Another File Finder
volume2/strings        Find text strings in files
volume2/curses/part01       Screen updating package
volume2/curses/part02
volume2/sozobon1.2/part01       Update to Sozobon C compiler
volume2/sozobon1.2/part02
volume2/sozobon1.2/part03
volume2/sozobon1.2/part04
volume2/sozobon1.2/part05
volume2/sozobon1.2/part06
volume2/sozobon1.2/part07
volume2/sozobon1.2/part08
volume2/sozobon1.2/part09


From -6271441004293742940
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,ee1035a0b86967e9,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,ee1035a0b86967e9,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-31 21:48:48 PST
Path: gmdzi!unido!mcsun!uunet!samsung!brutus.cs.uiuc.edu!apple!sun-barr!newstop!
sun!sunfedcomm!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st,comp.sys.atari.st
Subject: v03INF1: Introduction to comp.sources.atari.st
Message-ID: <34651@grapevine.uucp>
Date: 31 Oct 89 21:08:10 GMT
Expires: 28 Feb 90 21:08:09 GMT
Sender: daemon@sunfedcomm
Lines: 148
Approved: koreth@panarthea.ebay.sun.com (Steven Grimm)
Xref: gmdzi comp.sources.atari.st:244 comp.sys.atari.st:18072
Posted: Tue Oct 31 22:08:10 1989

Submitted-by: koreth@panarthea.ebay.sun.com (Steven Grimm)
Posting-number: Volume 3, Info 1
Archive-name: intro

This is the first of two introductory articles about comp.sources.atari.st.
This one describes how to submit source to the newsgroup, where the
archive sites are, and how to contact them.  The companion article lists
all previously-published sources.

I am always looking for suggestions on how to improve the usefulness
of the newsgroup, and can be contacted as listed below.

-- Steven Grimm
   koreth@panarthea.ebay.sun.com
```

Subject: Submitting source for publication

Items intended for posting or queries and problem notes should be sent to atari-sources@panarthea.ebay.sun.com. If you are on a UUCP-only site, you can send them to sun!ebay!panarthea!atari-sources. In Europe, send your submissions to Jan-Hinrich Fessel, the European submoderator of the group, at unido!atari-sources (or, if you're a masochist, atari-sources@unido.informatik.uni-dortmund.de.)

If you want verification of arrival, so say in a cover note, or at the beginning of your submission, if it is small. I try to verify that a program works, and if I can't get it to work, I may hold up posting it for a couple of days. Please note that, except in rare cases, source without documentation will not be published. The backlog from receipt to posting varies from one to four weeks depending mostly on the set of submissions currently in my queue.

I prefer that you submit sources and binaries separately, as it's a pain to separate them, and such submissions are invariably placed on the back burner. Sources should be submitted in "shar" format, plaintext and split up into reasonable (less than 50K) chunks, unless you don't have access to a shar program, in which case "arc" is all right.

Please inform me of any special system requirements, and what development environment was used, so that I can properly test your submission and label it when I post it.

Subject: The structure of comp.sources.atari.st articles

Each posting in comp.sources.atari.st is called an "issue"; there are roughly 100 issues to a volume. The division is arbitrary and may vary. There are two types of articles in comp.sources.atari.st: sources and "information postings." They can be distinguished by the subject line:

  Subject: v03INF3: Index for Volume 1 and other info

This first word in the title identifies this as the third info posting of volume one. Similarly, the subject line shown below:

  Subject: v03i081: tossrc -- Source code to TOS

identifies this as the 81st source article in Volume 3. Large sources are broken up into smaller pieces, and have subject lines that look like this:

  Subject: v03i082: VMS clone in ST-LOGO, Part03/08

Certain information about the system configuration required to use the

program is given on the keywords line.

```
  Keywords: MWC, Megamax, 1meg, high
```

This means that the program compiles with Mark Williams or Megamax C, requires at least one meg of RAM and only runs in high resolution. Following is a list of keywords; new ones may be added as needed. They are mostly self-explanatory.

```
  Alcyon  - Alcyon C (included with Atari developer's kit)
  MWC         - Mark Williams C
  Megamax - Megamax C
  Laser       - Laser C
  Lattice - Lattice C
  C       - Any C compiler
  OSI         - OSI Personal Pascal
  BASIC       - ST BASIC (this is not a joke!)
  uuencode    - program is uuencoded (uudecode required to unpack)
  arc         - program is archived (arc required to unpack)
  high        - high resolution
  medium  - medium resolution
  low         - low resolution
  1meg        - needs 1 meg of RAM
```

The first few lines of an article are auxiliary headers that look like this:

```
  Submitted-by: david@bdt.UUCP (David Beckmeyer)
  Posting-number: Volume 3, Issue 80
  Archive-name: micro-rtx
```

The "Submitted by" is the author of the program. If you have comments about the sources published in comp.sources.atari.st this is the person to contact. When possible, this address is in domain form, otherwise it is a UUCP bang path relative to some major (backbone) site.

The second line repeats the volume/issue information for the aide of notes sites and automatic archiving programs.

The Archive-name is the "official" name of this source in the archive. Large postings will have names that look like this:

```
Archive-name: desktop/part01
```

Please try to use this name when requesting that sources be mailed to you. Also, note that the "part number" given in the title, and the archive name given in the auxiliary header need not be identical.

Since most archive sites run UNIX, articles are given UNIX-style filenames rather than ST-style filenames. I do try to limit filenames to eight characters whenever possible, though.

Subject: Reporting and tracking bugs and patches to postings

Most of the time, bugs and updates will be posted to comp.sys.atari.st.
Sometimes, "official" updates from the author will be posted submitted to
comp.sources.atari.st so that they are archived with the sources.

Since there is no easy way to apply source code patches on the ST, changes
to a program often result in a whole new version being posted.

To report bugs, contact the person listed in the Submitted-to header.
Often there is a contact address in a README file, too. I do not maintain
the sources I moderate, so don't send your bug reports to me.

Subject: Listing of archive sites in no particular order

1. Site panarthea is running an archive server, which automatically answers
   mail requests. Send mail to archive-server@panarthea.ebay.sun.com with the
   word "help" in the message body for more information. Both the sources
   and binaries groups are archived on panarthea.
2. Unido is the European archive site. It's running an archive server
   similar to panarthea's; send "help" to unido!archive-server for more
   information. Unido archives both the sources and binaries groups.

If you know of additional sites which reliably archive comp.sources.atari.st
(or comp.binaries.atari.st, for that matter), please tell me so I can add
them to the list.

From 1213853758307906575
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,d5569b4b061d976b,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-31 22:12:52 PST
Path: gmdzi!unido!mcsun!uunet!samsung!gem.mps.ohio-state.edu!apple!sun-barr!
newstop!sun!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i002: uurename -- Rename binary postings to work with uud
Keywords: shar, C
Message-ID: <34656@grapevine.uucp>
Date: 31 Oct 89 23:49:26 GMT
Sender: news@grapevine.EBay
Reply-To: sun!pyramid!actisb!federico (Federico Heinz)
Lines: 82
Approved: koreth@panarthea.ebay.sun.com
Posted: Wed Nov 1 00:49:26 1989

Submitted-by: sun!pyramid!actisb!federico (Federico Heinz)
Posting-number: Volume 3, Issue 2
Archive-name: uurename

  Here's a tiny utility I wrote to help unpacking files from an archive server. You certainly know the problem: 'uud' expects all files except the first one to have a certain name, and that name is always different from the one you chose, no matter what you chose. With this program it's over: just name the files as you choose, for instance 'part01', 'part02',..., then type 'uurename part*', and you are done. All files have now their correct names, except for the ones that start a file (those not starting with 'begin part...' but just 'begin <permission> <name>'). All you have to do now is run uud on these 'partNN' files. Of course, this works only for files encoded with Dumas uuencode, and I have found some weirdly packed files that cause trouble, but it works most of the time, and even when it fails it does some useful work. After all, it's pretty trivial.

  The executable is there also for people without Sozobon C or Dlibs. The source compiled also on our BSD machine, but you must change the directory delimiter from '\' to '/'.

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      uurename.c
#
sed 's/^X//' << 'SHAR_EOF' > uurename.c &&
X#include <stdio.h>
X
Xstatic char buffer[200];
X
Xstatic void buildname(target, source, name)
Xchar *target, *source, *name;
X{
X    char *nameTarget;
X
X    nameTarget = target;
X    while (*source)
X        if ((*(target++) = *(source++)) == '\\')
X            nameTarget = target;
X    strcpy(nameTarget, name);
X}
X
X
Xmain(argc, argv)
Xint argc;
Xchar *argv[];
X{
X    int i;
X    char newname[20];
X    FILE *fp;
X    char drive[4], path[128], file[10], ext[4];
X
X    for (i=1; i < argc; i++)
X    {
```

```
X        if ((fp = fopen(argv[i], "r")) == NULL)
X            fprintf(stderr, "Unable to open: %s.\n", argv[i]);
X        else
X        {
X            while (fgets(buffer, 200, fp) != NULL)
X            {
X                if (!strncmp(buffer, "begin ", 6))
X                {
X                    if (sscanf(buffer, "begin part %*s %s", newname) == 1)
X                    {
X                        buildname(buffer, argv[i], newname);
X                        if (rename(argv[i], buffer) != 0)
X                            fprintf(stderr, "Cannot move %s to %s.\n",
X                                    argv[i], buffer);
X                    }
X                    break;
X                }
X            }
X            fclose(fp);
X    }
X    }
X}
SHAR_EOF
chmod 0600 uurename.c || echo "restore of uurename.c fails"
exit 0
```

```
From 4675997726765958751
X-Google-Language: GERMAN,ASCII-7-bit
X-Google-Thread: 100493,1a0a62aa547d0085,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-31 22:12:59 PST
Path: gmdzi!unido!mcsun!uunet!samsung!gem.mps.ohio-state.edu!apple!sun-barr!
newstop!sun!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i003: uustuff -- More uud-ing tools
Keywords: shar, C
Message-ID: <34657@grapevine.uucp>
Date: 31 Oct 89 23:52:55 GMT
Sender: news@grapevine.EBay
Reply-To: uunet!gmdzi!focke (Stefan Focke)
Lines: 355
Approved: koreth@panarthea.ebay.sun.com
Posted: Wed Nov  1 00:52:55 1989

Submitted-by: uunet!gmdzi!focke (Stefan Focke)
Posting-number: Volume 3, Issue 3
Archive-name: uustuff

Hello net-world!

This archive contains some programs witch helps you to uudecode
```

files you got from the net on your UNIX-machine.

If you like it, send me a mail.

Stefan Focke   (focke@gmdzi.UUCP)

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      liesmich.txt
#      readme.txt
#      uu_cat.c
#      uu_rename.c
#
sed 's/^X//' << 'SHAR_EOF' > liesmich.txt &&
X
XBei meinen Bemuehungen ein paar Programme vom Netz auf meinen ST zu holen,
Xstiess ich immer wieder auf Probleme:
X
XDateien, die ich vom "news reader" aus auf Platte geschrieben
Xhatte, waren, da der in der Uebersichtsliste angezeigte Titel nicht mit
Xdem uebereinstimmt, der hinter dem 'begin' der ua?-Datei steht, mit
Xeinem falschen Namen versehen. Das fuehrt dann dazu, dass wenn spaeter
Xauf dem ST "uudecode" ausgefuehrt wird, Programme, die aus meherern
XTeilen bestehen, nicht mehr zusammengesetzt werden koennen. Stattdessen
Xmuessen die Dateien erst umbenannt werden.
X
XDie weiteren Aufgaben der beiliegenden Programme sind, das uudecode
Xbereits auf der Unix-Maschine durchzufuehren und die entstehenden ARC-Files
Xauf Fehler zu ueberpruefen.
X
X
X
XBenutzung der Programme:
X
X- Vorbereitung: Compilieren der beiden C-Quellen
X  /* Dies sind meine ersten C-Programme. Kleine Schoenheitsfehler,
X     die alten C-Freaks die Haare zu Berge stehen lassen, koennen
X     daher durchaus enthalten sein. */
X  cc -o uu_rename.exe uu_rename
X  cc -o uu_cat.exe uu_cat
X  chmod u+x uu_rename
X  chmod u+x uu_cat
X
X- Speichern der einzelnen Teile eines vom Netz kommenden Programmes
X  unter einem beliebigen Namen. Die Extension sollte jedoch die Form
X  "ua?" haben.
X
X- Aufruf von uu_rename
X  Es werden alle "ua?"-Dateien bearbeitet. In den Dateien wird der hinter
X  "begin" stehende Dateiname gesucht und die Datei dementsprechend
X  umbenannt. Falls dieser Name nicht auf "ua?" endet, wird "uaa" als
```

```
X  Endung verwendet. Vor dem Umbenennen wird noch getestet, ob es die
X  Datei in die umbenannt werden soll bereits gibt. Falls dem so ist,
X  wird dies angezeigt und das Umbenennen nicht durchgefuehrt.
X
X- Aufruf von uu_cat
X  Da das uudecode meines UNIX-Rechners keine Dateien zusammensetzen kann,
X  also nicht mit den include-Anweisungen am Ende einer "ua?"-Datei arbeiten
X  kann, fuegt dieses Programm die Teile, aus denen die Datei besteht
X  zusammen. Es entsteht eine ".enc"-Datei, die die fuer "uudecode"
X  notwendigen Teile beinhaltete, sowie ein ".txt"-Datei, die Information,
X  die in der ".uaa"-Datei vor "table" steht, beinhaltet.
X  Nach dem Zusammensetzen wird uudecode aufgerufen. Anschliessend
X  wird auch "arc" mit den Optionen -v und -t ausgefuehrt.
X
X
XViel Erfolg
X
X   Stefan Focke     (focke@gmdzi.UUCP)
SHAR_EOF
chmod 0600 liesmich.txt || echo "restore of liesmich.txt fails"
sed 's/^X//' << 'SHAR_EOF' > readme.txt &&
X
XI had some Problems trying to get programs from the net.
X
XFiles I saved from the "news reader" often have wrong names, so "uudecode"
Xon my ST was not able to include all the files it needed. I had to
Xrename some files.
X
XThe other things the programs do are to uudecode the files on the UNIX-
Xmachine an to test, if the "ARC"-files are ok.
X
X
XHow to use:
X
X- Compile the C-sources
X  /* These are my first C-programs. So it is possible, that
X     they are not optimized */
X  cc -o uu_rename.exe uu_rename.c
X  cc -o uu_cat.exe uu_cat.c
X  chmod u+x uu_rename
X  chmod u+x uu_cat
X
X- Save the files coming from the net to your UNIX-disk. The name that you
X  give to the file has the only restriction, that the extension must
X  be of the form ".ua?"
X
X- Execute uu_rename
X  This program changes the names of all "ua?"-files to the name, witch
X  stands behind "begin" in this file. If a file with the new name exists
X  the program prints a message.
X
X- Execute uu_cat
X  This program puts a multi-part-file (uaa, uab, ) together to a
```

```
X  ".enc"-file. The text befor "table" in the "uaa"-file is written
X  into a ".txt"-file.
X  Then the batch-files executes uudecode and "arc -v" and "arc -t"
X
X
XGood luck
X
X  Stefan Focke   (focke@gmdzi.UUCP)
SHAR_EOF
chmod 0600 readme.txt || echo "restore of readme.txt fails"
sed 's/^X//' << 'SHAR_EOF' > uu_cat.c &&
X#include <stdio.h>
X
X#define MAXLINE 200
X#define TRUE 0
X#define FALSE 225
X
Xmain()
X{   int len,Flag;
X    char line[MAXLINE],line2[MAXLINE];
X
X    /* Lesen der Dateinamen aus dem Standard-Input */
X    /* Jede Zeile ist ein Dateiname */
X    while ( fgets(line,MAXLINE,stdin) != NULL )  {
X          line[strlen(line)-1] = '\0';
X          BearbeiteDatei(line);
X    }
X}
X
X
XBearbeiteDatei(FileName)
Xchar FileName[MAXLINE];
X{
X    FILE *fin, *fout;
X    char line[MAXLINE],HelpStr[MAXLINE];
X    char NeuFile[MAXLINE];
X    char FileExt[8];
X    int  OpenError,FileFlag;
X
X    printf("\nOpen file %s",FileName); printf("\n");
X    fin = fopen(FileName,"r");
X    FileExt[0] = '\0';
X    strcat(FileExt,".uaa");
X    extract(FileName,0,strlen(FileName)-4,FileName);
X
X    HelpStr[0]='\0'; strcat(HelpStr,FileName);
X    strcat(HelpStr,".txt");
X    fout = fopen(HelpStr,"w");
X
X    OpenError = FALSE; FileFlag = FALSE;
X    while ( fgets(line,MAXLINE,fin) != NULL )   {
X        extract(line,0,5,HelpStr);
X        if ( FileFlag = FALSE &&
```

```
X             (strcmp(HelpStr,"table")==0 || strcmp(HelpStr,"begin")==0) ) {
X       FileFlag = TRUE;
X           fclose(fout);
X           HelpStr[0] = '\0';
X           strcat(HelpStr,FileName);
X           strcat(HelpStr,".enc");
X           fout = fopen(HelpStr,"w");
X         }
X
X         extract(line,0,3,HelpStr);
X         if ( strcmp(HelpStr,"end") == 0 )  {
X           fputs("end",fout);
X           break;
X         }
X
X         extract(line,0,7,HelpStr);
X         if ( strcmp(HelpStr,"include") == 0 )  {
X           /* Oeffnen der neuen Datei, lesen bis hinter den Header */
X           fclose(fin);
X           extract(line,8,14,HelpStr); HelpStr[strlen(HelpStr)] = '\0';
X           printf("          %s", HelpStr);
X           if ((fin = fopen(HelpStr,"r")) == NULL )  {
X              printf ("  not found\n");
X          OpenError = TRUE;
X              break;
X           }
X           printf("\n");
X           while ( fgets(line,MAXLINE,fin) != NULL )   {
X               extract(line,0,5,HelpStr);
X               if ( strcmp(HelpStr,"begin") == 0 )
X                  break;
X           }
X           fgets(line,MAXLINE,fin);
X         }
X
X         fputs(line,fout);
X
X     } /* end while */
X
X     fputs("\n",fout);
X     if (OpenError == FALSE)
X        fclose(fin);
X     fclose(fout);
X}
X
X
X/*****************************************************************/
X/**********************  extract  ********************************/
X/*****************************************************************/
X
X/* Die Funktion schneidet aus der Zeichenkette 'str' ab
X   Position 'Start' 'len' Buchstaben heraus. Das Resultat
X   wird in 'Result' zurueckgegeben.
```

```
X*/
X
Xextract(str,start,len,result)
Xchar str[MAXLINE];
Xchar result[MAXLINE];
Xint start,len;
X
X{   int i;
X    char temp[MAXLINE];
X
X    for (i=0; i<len  &&  str[start+i]!='\0'  &&  str[start+i]!='\n'; ++i)
X        temp[i]=str[i+start];
X    temp[i]='\0';
X    for (i=0; (result[i]=temp[i]) != '\0'; ++i);
X}
SHAR_EOF
chmod 0600 uu_cat.c || echo "restore of uu_cat.c fails"
sed 's/^X//' << 'SHAR_EOF' > uu_rename.c &&
X#include <stdio.h>
X
X#define MAXLINE 100
X#define TRUE 0
X#define FALSE 225
X
Xmain()
X{   int len,Flag;
X    char line[MAXLINE],line2[MAXLINE];
X
X    /* Lesen der Dateinamen aus dem Standard-Input */
X    /* Jede Zeile ist ein Dateiname */
X    while ( fgets(line,MAXLINE,stdin) != NULL )  {
X          line[strlen(line)-1] = '\0';
X          BearbeiteDatei(line);
X    }
X}
X
X
XBearbeiteDatei(filename)
Xchar filename[MAXLINE];
X{
X    FILE *fp, *ftest;
X    char line[MAXLINE],line2[MAXLINE];
X    char NeuFile[MAXLINE];
X    char FileExt[8];
X    int  NeuLaenge;
X
X    fp = fopen(filename,"r");
X    while ( fgets(line,MAXLINE,fp) != NULL )   {
X        extract(line,0,5,line2);
X        if (strcmp("begin",line2) == 0)  {
X           extract(line,6,4,line2);
X           if (strcmp("part",line2) == 0)
X              extract(line,13,12,NeuFile);
```

```
X           else
X              extract(line,10,12,NeuFile);
X
X           printf ("Old name : %12s",filename);
X       printf ("   new : %12s",NeuFile);
X           /* Wenn die zu verwendende Endung nicht .ua? ist, */
X           /* muss diese in .uaa umgesetzt werden.          */
X           NeuLaenge = strlen(NeuFile);
X           extract(NeuFile,NeuLaenge-4,3,FileExt);
X           if (strcmp(".ua",FileExt) != 0 )  {
X              extract(NeuFile,0,NeuLaenge-4,NeuFile);
X              strcat(NeuFile,".uaa");
X              printf("\n                      changed : %12s",NeuFile);
X           }
X
X           if (strcmp(filename,NeuFile) != 0)  {
X              if ((ftest = fopen(NeuFile,"r")) != NULL) {
X                 printf ("   new file existst\n");
X                 fclose(ftest);
X                 break;
X              }
X           }
X           printf ("\n");
X           rename (filename,NeuFile);
X           break;
X        }
X    }
X    fclose(fp);
X}
X
X
X
X/*****************************************************************/
X/***********************  getline  ***************************/
X/*****************************************************************/
X
Xgetline(fp,s,lim)
Xchar s[];
Xint lim;
XFILE *fp;
X{   int c,i;
X
X    for (i=0; i<lim-1  &&  (c=getc(fp))!=EOF  &&  c!='\n';  ++i)
X        s[i]=c;
X    if (c=='\n')  { s[i]=c; ++i; }
X    s[i]='\0';
X    return(i);
X}
X
X
X/*****************************************************************/
X/***********************  extract  ***************************/
X/*****************************************************************/
```

```
X
X/* Die Funktion schneidet aus der Zeichenkette 'str' ab
X   Position 'Start' 'len' Buchstaben heraus. Das Resultat
X   wird in 'Result' zurueckgegeben.
X*/
X
Xextract(str,start,len,result)
Xchar str[MAXLINE];
Xchar result[MAXLINE];
Xint start,len;
X
X{   int i;
X    char temp[MAXLINE];
X
X/*
X    printf("\n");
X    printf("extract: %s",str);
X    printf("  %d",start); printf("   %d",len);
X    printf("\n");
X*/
X    for (i=0; i<len  &&  str[start+i]!='\0'  &&  str[start+i]!='\n'; ++i)
X        temp[i]=str[i+start];
X    temp[i]='\0';
X    for (i=0; (result[i]=temp[i]) != '\0'; ++i);
X/*    result = ""; strcat(result,temp);
X/*    printf("Result %s",result); printf("\n"); */
X}
SHAR_EOF
chmod 0600 uu_rename.c || echo "restore of uu_rename.c fails"
exit 0
```

```
From -6204981372794592559
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,1b09c0803019c35,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-31 22:13:10 PST
Path: gmdzi!unido!mcsun!uunet!cs.utexas.edu!ginosko!samsung!usc!apple!sun-barr!
newstop!sun!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i004: uustuff2 -- Addendum to previous posting
Keywords: shar, C
Message-ID: <34660@grapevine.uucp>
Date: 1 Nov 89 00:01:15 GMT
Sender: news@grapevine.EBay
Reply-To: uunet.uu.net!gmdzi!focke (Stefan Focke)
Lines: 51
Approved: koreth@panarthea.ebay.sun.com
Posted: Wed Nov  1 01:01:15 1989

Submitted-by: uunet.uu.net!gmdzi!focke (Stefan Focke)
Posting-number: Volume 3, Issue 4
```

Archive-name: uustuff2

[Oops! I left these out of the last posting. -sg]

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      uu_cat
#      uu_rename
#
sed 's/^X//' << 'SHAR_EOF' > uu_cat &&
Xls -l *.uaa | cut -c46-99 | sort > inhalt
Xrm *.txt
Xrm *.arc
Xrm *.ARC
Xuu_cat.exe < inhalt
Xecho
Xecho
Xfor i in *.enc
X do
X   echo uudecode von : $i
X   uudecode $i
X done
Xfor i in *.arc *.ARC
X do
X   echo
X   echo Arc von : $i
X   echo
X   arc -v $i
X   echo
X   arc -t $i
X   echo
X   echo
X done
X
Xrm inhalt
Xrm *.enc
X
Xll *.ARC *.arc *.txt
SHAR_EOF
chmod 0600 uu_cat || echo "restore of uu_cat fails"
sed 's/^X//' << 'SHAR_EOF' > uu_rename &&
Xls -l *.u?? | cut -c46-99 | sort > inhalt
Xuu_rename.exe < inhalt
Xrm inhalt
SHAR_EOF
chmod 0600 uu_rename || echo "restore of uu_rename fails"
exit 0
```

From 852805802335733648
X-Google-Language: ENGLISH,ASCII-7-bit

X-Google-Thread: 100493,215a54ee6cfd62c3,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-10-31 22:31:00 PST
Path: gmdzi!unido!mcsun!uunet!lll-winken!brutus.cs.uiuc.edu!apple!sun-barr!
newstop!sun!sunfedcomm!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i001: fract -- Monochrome Mandelbrot generator
Keywords: shar, C, high
Message-ID: <34653@grapevine.uucp>
Date: 31 Oct 89 21:12:12 GMT
Sender: daemon@sunfedcomm
Reply-To: rosenkra@hall.cray.com (Bill Rosenkranz)
Lines: 1896
Approved: koreth@panarthea.ebay.sun.com
Posted: Tue Oct 31 22:12:12 1989

Submitted-by: rosenkra@hall.cray.com (Bill Rosenkranz)
Posting-number: Volume 3, Issue 1
Archive-name: fract

this is just something i wipped up one day. i don't remember ever seeing
a mono mandelbrot generator (i dislike color :-), so here goes...enjoy.

it uses line A to plot pixels. should be portable to any bitmap computer
(i.e. sun, apollo, sgi, amiga, mac, pc, X11, ...). i tried to make the
portability things seperated for relative ease. i plan on doing an
X11 port soon (on a cray-2, to get some SPEED). i'd like to be able to
generate and display an image every few seconds (network is the limiting
factor :^). yafttms (yet another fly thru the mandelbrot set)...i will
also change the file to use the RLE-like scheme i mention somewhere...

in the spirit of a recent c.lang.c thread, the program uses some goto's!
included are some atari-specific line-A calls (like MWC libraries, i
think) which might be of general interest. written in alcyon as68.
i did not spend any time at all on the algorithm so expect 500+ second
runs. use "-i 50" for most everything (unless u run overnite). no
mouse support (you specify coord on command line).

-bill rosenkranz 10/2/89
rosenkra%boston@hall.cray.com

(binaries in c.b.atari.st)

------- cut here -------------------------
```
#! /bin/sh
# This is a shar archive.  Extract with sh, not csh.
echo x - Readme
cat > Readme << '22842!Funky!Stuff!'
NAME
     fract - Generate monochrome Mandelbrot set

USAGE
```

```
    fract [options]

       -i iterations (1000)         defines "convergence"
       -x xcenter (-1.0)        center of viewport
       -y ycenter (0.0)
       -r range (3.0)           size of viewport (width=height)
       -s scale (3)             0 to 3, 0=coarse 3=fine
       -v                   invert colors
       -a                   add contours
       -h                   hold screen before exit
       -o outfile           write results
       -f infile            view existing file (ignores -i,-x,-y,
                      -r,-s)

[defaults in parentheses. more help in man page, fract.6  -wjr]

this is a monochrome mandelbrot generator. it was originally written for
atari ST, but should port easily to amiga, pc, mac, sun, and X11. all you
really need is to be able to address and color a pixel. it may work on
color (med rez, i have not tried it) but you will only get the top half of
the image.

it generates images in a square only. on the st, this is 360x360 pixels
or about 129,000 pixels. it can take a while to generate an image so
it does it in 2 passes. the first pass is very coarse (45x45 "big" pixels).
you can break out of the calculation at any time with any keystroke.
you can also dump the binary data to a file (probably not portable for
non-680x0 systems because of byte order of shorts).

it works from a command line. you can set the iterations limit (default
is 1000 - much to large for most hacking). try "-i 50" for most work.
the default scaling is for finest image (single pixel). for a quicker
scan, try "-s 2" (arg for -s is 0 to 3, 0 is coarse, 3 is fine).

there are a couple of files to play with. these are binary data files
and unarc to around 250kB each. they are setup for atari ST using ffp
floats. there are only 3 floats stored and they are only used for annotation
so just skip them if you like. the files are in M0.ARC and M1.ARC. m0
is the full Mandelbrot set and m1 is a blowup of the area near the
tangency of the 2 big circles in the set. generate them with:

    fract -o m0 -i 50
    fract -o m1 -i 50 -x -0.9 -y -0.3 -r 0.13

Feel free to copy. Leave this file alone! No other restrictions on copying
or hacking. Use as you like.


(c) 1989 Bill Rosenkranz

net: rosenkra%boston@hall.cray.com
CIS: 71460,17
GENIE:   W.ROSENKRANZ
```

```
22842!Funky!Stuff!
echo x - Manifest
cat > Manifest << '22842!Funky!Stuff!'
total 42
-rw-r--r--  1 rosenkra      526 Oct  2 10:33 Makefile
-rw-r--r--  1 rosenkra      601 Oct  2 17:09 Manifest
-rw-r--r--  1 rosenkra     2242 Oct  2 10:33 Readme
-rw-r--r--  1 rosenkra     4667 Oct  2 17:07 fract.6
-rw-r--r--  1 rosenkra    20973 Oct  2 10:33 fract.c
-rw-r--r--  1 rosenkra     5706 Oct  2 17:03 fract.man
-rw-r--r--  1 rosenkra      690 Oct  2 10:33 linea0.s
-rw-r--r--  1 rosenkra      774 Oct  2 10:33 linea1.s
-rw-r--r--  1 rosenkra      685 Oct  2 10:33 linea2.s
-rw-r--r--  1 rosenkra     1168 Oct  2 10:33 linea3.s
-rw-r--r--  1 rosenkra      155 Oct  2 10:33 version.h
22842!Funky!Stuff!
echo x - fract.man
cat > fract.man << '22842!Funky!Stuff!'
```


NAME

fract - draw Mandelbrot fractals

SYNOPSIS

fract [options]

DESCRIPTION

Fract generates, stores, and displays fractal images from the Mandelbrot set. Fract was originally written for the Atari ST but it should be very easy to port to other hosts like Macintosh, Amiga, PC, Sun, and X. All you really need is to address a pixel and color it. It is meant for monochrome displays but can be made to work on color systems as well. On the Atari ST, it is meant for a shell but also works from the desktop.

Fract was designed to vary both the threshhold for convergence (or divergence) as well as the resolution of the display. The default is maximum for both which can take a long time to process. For most cases, you should be able to get by with iteration count of 50 to 100. The default coordinates plots the entire set. Use -x, -y, and -r to zoom in. Other default values will simply plot the values in the set black, outside white. The -a option allows you to alternate black and white contours for those values outside the set

(where color displays make the pretty pictures).

For resolutions higher than the lowest (-s 0), fract first calculates and plots the image for -s 0 -i 50, then calculates and plots the desired image. The first image is fairly fast on the Atari ST (about 30 seconds) and looks like "fat" pixels 8x8. This allows you to break the calculation if the image is not in range (with any key press). On non-unix systems with no interrupt signals, you can break the calculation with a keystroke after any pixel is done.

This program was based on an article in the December 1986 BYTE.

OPTIONS

The following command-line options are available:

-a Causes every other contour value to be drawn black. The default is to just plot the threshhold (-i value) and above.

-i iter

The default number of iterations dividing values in the set versus values outside the set. The default is 1000 though 50 or 100 should be sufficient for most things.

-x xcenter

The x coordinate of the viewing window. The default is -1.0.

-y ycenter

The y coordinate of the viewing window. The default is 0.0.

-r range

The x and y range of the viewing window. The default is 3.0.

-s scale
The scale factor. Possible values currently are 0, 1, 2, and 3. 0 is for coarse and 3 is for fine grain.

-v This toggles the "colors" (black to white and visa versa).

-h Hold display after plot (before return to the Atari desktop, for example).

-o outfile
Write results to file outfile.

-f infile
Display previous results from file infile.

PORTING

Fract was designed to be ported to other environments. Most of the code follows "normal" Unix conventions. The file I/O needs changing for porting to non-680x0 architectures, though it has even run on a Cray!

The following funtions are non-portable:

```
int  init ()       - initialize things
int  finish ()     - finish up
int  check_key ()  - check for a key press (interrupt)
int  wait_key ()   - wait for a key press
int  do_pixel ()   - plot a pixel of some color
int  do_line ()    - draw a line
int  cursor ()     - toggle cursor on/off
int  mv_cursor ()  - move cursor on screen
int  clr_screen () - clear screen, home cursor
long do_time ()    - for elapsed times
```

FILE FORMAT

The binary file format for saved images uses non-portable

formats (no binary data file is portable). Binary data is much more compact, however. It is a simple matter to write a simple program to convert to ASCII, however.

In general, words are 16-bit. Most of the data is integer words. The byte order is normal 68000 order (first byte read is most significant byte of word). The floating point data was originally using the Motorola Fast Floating Point format (FFP) largely because the development compiler is twice as fast with FFP versus IEEE. Note that the floating point data in the file is only for the textural display on the image and will not effect image display at all.

Here is the data format:

```
magic           (8 char, !<fract>)
machine         (1 16-bit word, 0=Atari, 1=Amiga, 2=PC,
                 3=Mac, 4=Sun,...)
version         (1 16-bit word, e.g. 100 or 0x00C8)
iter            (1 16-bit word)
scale           (1 16-bit word, 45, 90, 180, 360 pixels
                 per unit)
x               (1 32-bit FFP float)
y               (1 32-bit FFP float)
range           (1 32-bit FFP float)
data            (scale * scale 16-bit words)
...
```

AUTHOR

Version 1.0 October 1989 (c) 1989 Bill Rosenkranz

net: rosenkra%boston@hall.cray.com
CIS: 71460,17
GENIE: W.ROSENKRANZ

```
22842!Funky!Stuff!
echo x - fract.6
cat > fract.6 << '22842!Funky!Stuff!'
.TH FRACT 6
.SH NAME
fract - draw Mandelbrot fractals
.SH SYNOPSIS
fract [options]
.SH DESCRIPTION
Fract generates, stores, and displays fractal images from the Mandelbrot set.
Fract was originally written for the Atari ST but it should be very easy
to port to other hosts like Macintosh, Amiga, PC, Sun, and X.
All you really need is to address a pixel and color it.
It is meant for monochrome displays but can be made to work on color
systems as well.
On the Atari ST, it is meant for a shell but also works from the desktop.
.PP
Fract was designed to vary both the threshhold for convergence (or divergence)
as well as the resolution of the display.
The default is maximum for both which can take a long time to process.
For most cases, you should be able to get by with iteration count of 50 to 100.
The default coordinates plots the entire set. Use -x, -y, and -r to zoom in.
Other default values will simply plot the values in the set black, outside
white.
The -a option allows you to alternate black and white contours for those
values outside the set (where color displays make the pretty pictures).
.PP
For resolutions higher than the lowest (-s 0), fract first calculates and
plots the image for -s 0 -i 50, then calculates and plots the desired image.
The first image is fairly fast on the Atari ST (about 30 seconds) and looks
like "fat" pixels 8x8.
This allows you to break the calculation if the image is not in range (with
any key press).
On non-unix systems with no interrupt signals, you can break the calculation
with a keystroke after any pixel is done.
.PP
This program was based on an article in the December 1986 BYTE.
.SH OPTIONS
The following command-line options are available:
```

.IP -a
Causes every other contour value to be drawn black. The default is to just plot the threshhold (-i value) and above.

.IP "-i iterations"
The default number of iterations dividing values in the set versus values outside the set. The default is 1000 though 50 or 100 should be sufficient for most things.

.IP "-x xcenter"
The x coordinate of the viewing window. The default is -1.0.

.IP "-y ycenter"
The y coordinate of the viewing window. The default is 0.0.

.IP "-r range"
The x and y range of the viewing window. The default is 3.0.

.IP "-s scale"
The scale factor. Possible values currently are 0, 1, 2, and 3. 0 is for coarse and 3 is for fine grain.

.IP -v
This toggles the "colors" (black to white and visa versa).

.IP -h
Hold display after plot (before return to the Atari desktop, for example).

.IP "-o outfile"
Write results to file outfile.

.IP "-f infile"
Display previous results from file infile.

.SH PORTING
Fract was designed to be ported to other environments. Most of the code follows "normal" Unix conventions. The file I/O needs changing for porting to non-680x0 architectures, though it has even run on a Cray!
.PP
The following funtions are non-portable:
.nf

```
int  init ()       - initialize things
int  finish ()     - finish up
int  check_key ()  - check for a key press (interrupt)
int  wait_key ()   - wait for a key press
int  do_pixel ()   - plot a pixel of some color
int  do_line ()    - draw a line
int  cursor ()     - toggle cursor on/off
int  mv_cursor ()  - move cursor on screen
int  clr_screen () - clear screen, home cursor
long do_time ()    - for elapsed times
```

.fi

```
.SH FILE FORMAT
The binary file format for saved images uses non-portable formats (no binary
data file is portable).
Binary data is much more compact, however.
It is a simple matter to write a simple program to convert to ASCII, however.
.PP
In general, words are 16-bit. Most of the data is integer words. The byte
order is normal 68000 order (first byte read is most significant byte of
word).
The floating point data was originally using the Motorola Fast Floating
Point format (FFP) largely because the development compiler is twice as
fast with FFP versus IEEE.
Note that the floating point data in the file is only for the textural
display on the image and will not effect image display at all.
.PP
Here is the data format:
.nf

magic           (8 char, !<fract>)
machine         (1 16-bit word, 0=Atari, 1=Amiga, 2=PC,
                 3=Mac, 4=Sun,...)
version         (1 16-bit word, e.g. 100 or 0x00C8)
iter            (1 16-bit word)
scale           (1 16-bit word, 45, 90, 180, 360 pixels
                 per unit)
x               (1 32-bit FFP float)
y               (1 32-bit FFP float)
range           (1 32-bit FFP float)
data            (scale * scale 16-bit words)
\&...
.fi

.SH AUTHOR
.nf
Version 1.0 October 1989 (c) 1989 Bill Rosenkranz

net:    rosenkra%boston@hall.cray.com
CIS:    71460,17
GENIE:  W.ROSENKRANZ
.fi
22842!Funky!Stuff!
echo x - Makefile
cat > Makefile << '22842!Funky!Stuff!'
# makefile for atari TOS version of fract (alcyon 4.14)
#
# [flags are for my custom cc(1) and make(1). yours may vary.  -wjr]

FP_FLAG       =             # use -e for ieee
CFLAGS        = -new             # -new for alcyon 4.14
LDFLAGS       = -new -s $(FP_FLAG)   # -s for no symbols
LIBS     =
FR_OBJS       = fract.o linea0.o linea1.o linea3.o
```

```
TARGET        = fract.ttp



$(TARGET):    $(FR_OBJS)
         $(CC) $(LDFLAGS) -o $(TARGET) $(FR_OBJS) $(LIBS)



fract.o:  fract.c version.h

linea0.o: linea0.s
         $(CC) -c $(CFLAGS) linea0.s

linea1.o: linea1.s
         $(CC) -c $(CFLAGS) linea1.s

linea3.o: linea3.s
         $(CC) -c $(CFLAGS) linea3.s

22842!Funky!Stuff!
echo x - fract.c
cat > fract.c << '22842!Funky!Stuff!'
#define atarist        /* machine (atarist, amiga, ibmpc, mac, sun, x11) */
#define alcyon         /* specific compiler, if any */


/*
 *   NAME:         fract - Generate monochrome Mandelbrot set
 *
 *   AUTHOR:       (c) 1989 Bill Rosenkranz
 *
 *            net: rosenkra%boston@hall.cray.com
 *            CIS: 71460,17
 *            GENIE:   W.ROSENKRANZ
 *
 *            Feel free to copy. Leave this comment alone! No other
 *            restrictions on copying or hacking. Use as you like.
 *
 *   MOTIVATION:   I don't ever remember seeing a monochrome fractal
 *            generator. I like this "zebra" effect with -a. Try:
 *
 *                 fract -a -i 50 -x -0.9 -y 0.3 -r 0.5
 *
 *            or unpack m1.arc and do:
 *
 *                 fract -a -f m1
 *
 *   COMPILE (ST): cc -o fract.ttp fract.c linea0.s linea1.s linea3.s
 *
 *   PORTABILITY:  Works with Alcyon 4.14 (Atari ST), should port easily
 *            to Mac, Amiga, Sun, PC, X, etc. except for file
 *            formats on non-680x0 architectures.
```

```
*
*           Floats in the files are only for text annotation.
*           Uses FFP floats not IEEE in files (3 numbers, stored
*           with putl). FFP is almost 2x faster. Here are formats
*           for FFP and IEEE:
*
*           FFP binary representation of 1.000000 is:
*
*              /---------------------------- mantissa (24 bits)
*              |                    /------- sign (1 bit)
*              |                    |   /--- exponent (7 bits)
*              |                    |   |
*           mmmmmmmmmmmmmmmmmmmmmmmmseeeeeee
*           10000000000000000000000001000001
*           ^
*           MSbit
*
*           mantissa: 10000000000000000000000
*           sign:     0
*           exponent: 1000001
*
*
*           IEEE binary representation of 1.000000 is:
*
*           /----------------- sign (1 bit)
*           |   /------------- exponent (8 bits)
*           |   |         /--- mantissa (23 bits)
*           |   |         |
*           seeeeeeeemmmmmmmmmmmmmmmmmmmmmmm
*           00111111100000000000000000000000
*           ^
*           MSbit
*
*           sign:     0
*           exponent: 01111111(127)
*           mantissa: 00000000000000000000000(0)
*
*
*           Note: getw() must read 16-bit int and getl() must
*           read 32-bit int from stream. putw() and putl() should
*           work with the same sizes, respectively.
*
*           For anyone hacking on the file stuff, do us all a
*           favor and not try to write structures to the file.
*           binary is bad enough. Binary structures is impossible.
*
*   NOTES:       Rough timings for scale=FINE, default x,y,range
*           (8 Mhz atari ST):
*
*           -i   time
*           -----    -------
*           10   560 sec
*           50   1350 sec
```

```
 *            100  2280 sec
 *            (-i 100 on a cray-2 is about 2 sec, no hand opt *8^)
 *
 *            The more points in the set in the zoom range, the
 *            longer it takes because it has to go to max_iter.
 *            The algorithm is pretty week and could be beefed up.
 *            This was a one day hack.
 *
 *            I was thinking of some sort of RLE-like encoding of
 *            data. How about if MSbit set, that word tells now
 *            many of the next word gets repeated? So rather than:
 *
 *                 .... 0001 0001 0001 0001 0001 0001 0005 ....
 *
 *            you use:
 *
 *                 .... 8006 0001 0005 ....
 *
 *            This takes much longer to read and write a file.
 *
 *            Make as a .ttp and use -h to hold for desktop or to
 *            take a snapshot.
 */

#include <stdio.h>
#include <math.h>
#include "version.h"


/*
 *   for portability of binary file (ha!)
 */
#undef WORD
#define WORD       int       /* 16-bit required */
#undef REAL
#define REAL       float         /* 32-bit required */
#undef LONG
#define LONG       long      /* 32-bit required */
#undef BYTE_WORD
#define BYTE_WORD  2         /* bytes per word */


#define BLACK      1         /* pixel colors */
#define WHITE      0

#define COARSE          45        /* ok for mono ST (pixels size) */
#define MED_COARSE 90        /* 360x360 pixel display */
#define MED_FINE   180
#define FINE       360

#define F_ATARIST  0         /* for file formats. all 680x0 */
#define F_AMIGA         1         /* should be the same. */
#define F_IBMPC         2
```

```
#define F_MACINTOSH    3
#define F_SUN      4


/*
 *   forward references and externals. non-portable are indicated...
 */
extern long   getl ();
extern int    getw ();      /* MUST read 16-bit (short) */

int      do_mandel ();
int      plt_mandel ();
int      plot_logic ();
int      pr_params ();
int      usage ();
int      init ();      /* atari ST */
int      finish ();         /* atari ST */
int      check_key ();      /* atari ST (BIOS) */
int      wait_key ();       /* atari ST (BIOS) */
int      do_pixel ();       /* atari ST (line A) */
int      do_line ();        /* atari ST (line A) */
int      cursor ();         /* atari ST (gemdos, vt52) */
int      mv_cursor ();      /* atari ST (vt52) */
int      clr_screen ();          /* atari ST (gemdos, vt52) */
long     do_time ();        /* atari ST (gemdos, sysvar) */


WORD     array_G[FINE][FINE];   /* storage for the set (~250 kB) */



/*------------------------------*/
/*   main          */
/*------------------------------*/
main (argc, argv)
int  argc;
char   *argv[];
{
     int      max_iter;/* the threshold (normally 1000) */
     int      scale;        /* pixels per unit */
     int      oset;         /* offset of plot from scrn corner */
     int      normcolor;    /* 1=normal, 0=reverse colors */
     int      alternate;    /* 0=no contours, 1=use alt cont */
     int      hold_scrn;    /* for hold before return to desktop */
     int      save_it; /* to save as file */
     int      load_it; /* to load from file */
     int      ret;
     int      this_vers = VERSION;
     int      file_vers;
     int      this_format = F_ATARIST;
     int      file_format;
     long     elapsed; /* elapsed seconds */
     float         x,       /* coord of view center */
```

```
          y;
float          range;        /* view width,height */
char      fname[256];
char      this_magic[10];
char      file_magic[10];    /* magic for file */
FILE        *stream;
union F_TO_L            /* for float<-->long (i/o only) */
{
          LONG _l;
          REAL _f;

}         f_to_l;



/*
 *   set defaults, initialize...
 */
strcpy (this_magic, "!<fract>");
normcolor = 1;
alternate = 0;
max_iter  = 1000;
scale     = FINE;
oset      = 20;
hold_scrn = 0;
save_it   = 0;
load_it   = 0;

x         = -0.5;       /* image of full set... */
y         =  0.0;
range     =  3.0;

init ();



/*
 *   parse args...
 */
argc--, argv++;
while (argc)
{
     if (**argv == '-')
     {
          switch (*(*argv+1))
          {
          case 'i':          /* iterations */
          case 'I':
               argc--, argv++;
               if (argc < 1)
                    usage ();
               max_iter = atoi (*argv);
               if (max_iter < 1 || max_iter > 1000)
```

```
          max_iter = 1000;
     break;

case 'x':          /* x coord of center */
case 'X':
     argc--, argv++;
     if (argc < 1)
          usage ();
     x = atof (*argv);
     break;

case 'y':          /* y coord of center */
case 'Y':
     argc--, argv++;
     if (argc < 1)
          usage ();
     y = atof (*argv);
     break;

case 'r':          /* range */
case 'R':
     argc--, argv++;
     if (argc < 1)
          usage ();
     range = atof (*argv);
     break;

case 's':          /* scale */
case 'S':
     argc--, argv++;
     if (argc < 1)
          usage ();
     scale = atoi (*argv);
     switch (scale)
     {
     case FINE:
     case COARSE:
     case MED_FINE:
     case MED_COARSE:
          break;
     case 0:
          scale = COARSE;
          break;
     case 1:
          scale = MED_COARSE;
          break;
     case 2:
          scale = MED_FINE;
          break;
     default:
          scale = FINE;
          break;
     }
```

```
               break;

          case 'v':          /* reverse video */
          case 'V':
               normcolor = 0;
               break;

          case 'a':          /* alternate even/odd */
          case 'A':
               alternate = 1;
               break;

          case 'h':          /* hold screen */
          case 'H':
               hold_scrn = 1;
               break;

          case 'o':          /* save */
          case 'O':
               argc--, argv++;
               if (argc < 1)
                    usage ();
               strcpy (fname, *argv);
               save_it = 1;
               break;

          case 'f':          /* load file */
          case 'F':
               argc--, argv++;
               if (argc < 1)
                    usage ();
               strcpy (fname, *argv);
               load_it = 1;
               break;

          default:
               usage ();
          }
     }
     else
          usage ();

     argc--, argv++;
}

if (alternate)
     normcolor = (normcolor) ? 0 : 1;




/*
 *   if loading, read in data...
```

```
     */
    if (load_it)
    {
         /*
          *   file format (words are 16-bit):
          *
          *   magic         (8 bytes)
          *   file format   (word)
          *   program version    (word)
          *   iter          (word)
          *   scale         (word)
          *   x        (32-bit ffp float)
          *   y        (32-bit ffp float)
          *   range         (32-bit ffp float)
          *   data          (scale * scale 16-bit words)
          *
          *   (yes, i know binary data is NOT portable...)
          *
          *   BTW: the union is used because a cast to long would
          *   convert (say) 1.789 to 1. the union unsures we store
          *   it with bits unchanged...
          */
         int  i;
         int  j;

#ifdef alcyon
         if ((stream = fopenb (fname, "r")) == (FILE *) 0)
#else
         if ((stream = fopen (fname, "rb")) == (FILE *) 0)
#endif
         {
              mv_cursor (51, 22);
              fprintf (stderr,
                   "open input err, %s",
                   fname);

              goto xit;
         }

         for (i = 0; i < 8; i++)
              file_magic[i] = (char) getc (stream);

         file_format = getw (stream);
         file_vers   = getw (stream);
         max_iter    = getw (stream);
         scale       = getw (stream);
         f_to_l._l   = (LONG) getl (stream); x     = (float) f_to_l._f;
         f_to_l._l   = (LONG) getl (stream); y     = (float) f_to_l._f;
         f_to_l._l   = (LONG) getl (stream); range = (float) f_to_l._f;
         for (i = 0; i < scale; i++)
         {
#ifdef alcyon
              fread (array_G[i], BYTE_WORD, scale, stream);
```

```
/*            read (stream->_fd, array_G[i], scale * 2);
              this was significantly fast but did not work
              properly. hmmm...*/
#else
              for (j = 0; j < scale; j++)
                   array_G[i][j] = (WORD) getw (stream);
#endif
          }

          fclose (stream);

          /*
           *   print parameters...
           */
          pr_params (max_iter, scale, oset, x, y, range);

          /*
           *   ...and just plot the set...
           */
          plt_mandel (max_iter, scale, oset, normcolor, alternate, x, y, range);

          goto xit;
     }



     /*
      *   this could eventually be a loop to allow users to pick new
      *   zoom area (here to xit:)
      */


     /*
      *   print parameters...
      */
     pr_params (max_iter, scale, oset, x, y, range);



     /*
      *   do coarse grain (if interupted, proceed to fine grain)...
      */
     do_time (1);

     ret = do_mandel (((max_iter > 50) ? 50 : max_iter), COARSE, oset,
               normcolor, alternate, x, y, range);

     elapsed = do_time (2);
     mv_cursor (51, 18);    printf ("time    = %ld (sec)", elapsed);



     /*
```

```
     *   ...now do fine grain...
     */
    if (scale != COARSE)
    {
        do_time (1);

        ret = do_mandel (max_iter, scale, oset, normcolor, alternate,
                 x, y, range);

        elapsed = do_time (2);

        mv_cursor (51, 18);  printf ("time    = %ld (sec)", elapsed);

        if (ret)
        {
            mv_cursor (51, 22);    printf ("aborted!");

            goto xit;
        }
    }



    /*
     *   save file if needed...
     */
    if (save_it)
    {
        int  i;
        int  j;

#ifdef alcyon
        if ((stream = fopenb (fname, "w")) == (FILE *) 0)
#else
        if ((stream = fopen (fname, "wb")) == (FILE *) 0)
#endif
        {
            mv_cursor (51, 22);
            fprintf (stderr,
                 "open output err, %s\n",
                 fname);

            goto xit;
        }

        for (i = 0; i < 8; i++)
            putc (this_magic[i], stream);
        putw (this_format, stream);
        putw (this_vers,   stream);
        putw (max_iter,    stream);
        putw (scale,       stream);
        f_to_l._f = (REAL) x;      putl ((long) (f_to_l._l), stream);
        f_to_l._f = (REAL) y;      putl ((long) (f_to_l._l), stream);
```

```
          f_to_l._f = (REAL) range;   putl ((long) (f_to_l._l), stream);
          for (i = 0; i < scale; i++)
          {
#ifdef alcyon
               fwrite (array_G[i], BYTE_WORD, scale, stream);
#else
               for (j = 0; j < scale; j++)
                    putw (array_G[i][j], stream);
#endif
          }

          fclose (stream);
     }




xit:
     /*
      *   clean up and exit (no return)
      */
     finish (hold_scrn);
}




/*-----------------------------*/
/*   usage              */
/*-----------------------------*/
usage ()
{
     fprintf (stderr, "fract [options]\n");
     fprintf (stderr, "\n");
     fprintf (stderr, "        -i iterations (1000)\n");
     fprintf (stderr, "        -x xcenter (-1.0)\n");
     fprintf (stderr, "        -y ycenter (0.0)\n");
     fprintf (stderr, "        -r range (3.0)\n");
     fprintf (stderr, "        -s scale (360 or 3)\n");
     fprintf (stderr, "        -v\n");
     fprintf (stderr, "        -a\n");
     fprintf (stderr, "        -h\n");
     fprintf (stderr, "        -o outfile\n");
     fprintf (stderr, "        -f infile\n");

     cursor (1);
     exit (1);
}




/*-----------------------------*/
```

```
/*   pr_params     */
/*-----------------------------*/
pr_params (max_iter, scale, oset, x, y, range)
int  max_iter;
int  scale;
int  oset;
float    x;
float    y;
float    range;
{

/*
 *   print parameters off to the side and make a box for the set.
 */

    int  l_scale = FINE;
    int  x1, x2, y1, y2;

    mv_cursor (49, 1); printf ("%s", version_G);

    mv_cursor (51, 3); printf ("iter    = %d", max_iter);
    mv_cursor (51, 4); printf ("xcenter = %f", x);
    mv_cursor (51, 5); printf ("ycenter = %f", y);
    mv_cursor (51, 6); printf ("range   = %f", range);
    mv_cursor (51, 7); printf ("scale   = %d", scale);

    mv_cursor (51, 9); printf ("xmin    = %f", x - range/2.0);
    mv_cursor (51, 10);    printf ("xmax    = %f", x + range/2.0);
    mv_cursor (51, 11);    printf ("ymin    = %f", y - range/2.0);
    mv_cursor (51, 12);    printf ("ymax    = %f", y + range/2.0);

    fflush (stdout);

    /*
     *   draw box...
     */
    x1 = oset-1;       x2 = oset+l_scale;     /* top */
    y1 = oset-1;       y2 = y1;
    do_line (x1, y1, x2, y2);

    x1 = oset-1;       x2 = oset+l_scale;     /* bottom */
    y1 = oset+l_scale; y2 = y1;
    do_line (x1, y1, x2, y2);

    x1 = oset-1;       x2 = x1;           /* left */
    y1 = oset-1;       y2 = oset+l_scale;
    do_line (x1, y1, x2, y2);

    x1 = oset+l_scale; x2 = x1;           /* right */
    y1 = oset-1;       y2 = oset+l_scale;
    do_line (x1, y1, x2, y2);
}
```

```
/*------------------------------*/
/*   do_mandel     */
/*------------------------------*/
int  do_mandel (iter, scale, oset, normcolor, alternate, x, y, range)
int  iter;
int  scale;
int  oset;
int  normcolor;
int  alternate;
float    x,
     y;
float    range;
{

/*
 *   calculate the set and plot it as it is generated
 */

     register int  max_iter = iter;
     register int  ix,       /* pixel number */
               iy;       /* row number */
     register int  count;        /* count is the contour level */
     float         size;         /* thing to compare for convergence */
     float         a,        /* intermediates... */
               b,
               ac,
               bc,
               b1;
     float         gap;      /* units for each pixel */



     gap =  range / (float) scale;


     /*
      *   each row...
      */
     for (iy = 0; iy < scale; iy++)
     {
          bc = y + (range / 2.0) - (float) iy * gap;

          /*
           *   each pixel in row...
           */
          for (ix = 0; ix < scale; ix++)
          {
               ac    = x - (range / 2.0) + (float) ix * gap;
               a     = ac;
               b     = bc;
```

```
                size  = 0.0;
                count = 0;

                /*
                 *   iterate...
                 */
                while (size < 4.0 && count < max_iter)
                {
                     b1   = 2 * a * b;
                     a    = a * a - b * b + ac;
                     b    = b1 + bc;
                     size = a * a + b * b;

                     /*
                      *   this is what we are after: the number
                      *   of iterations to convergence/divergence
                      */
                     count++;
                }


                /*
                 *   store it...
                 */
                array_G[ix][iy] = (WORD) count;


                /*
                 *   if in set, plot pixel
                 */
                plot_logic (normcolor, alternate, count, max_iter,
                     scale, ix, iy, oset);


                /*
                 *   was a key pressed? if so, exit for now...
                 */
                if (check_key ())
                     return (1);
           }
      }
      return (0);
}



/*------------------------------*/
/*   plt_mandel         */
/*------------------------------*/
plt_mandel (iter, scale, oset, normcolor, alternate, x, y, range)
int  iter;
int  scale;
```

```
int  oset;
int  normcolor;
int  alternate;
float    x,
     y;
float    range;
{

/*
 *   just plot a set in global array_G
 */

     register int  max_iter = iter;
     register int  ix,       /* pixel number */
               iy;       /* row number */
     register int  count;


     /*
      *   each row...
      */
     for (iy = 0; iy < scale; iy++)
     {
          /*
           *   each pixel in row...
           */
          for (ix = 0; ix < scale; ix++)
          {
               /*
                *   get count from array...
                */
               count = (int) array_G[ix][iy];


               /*
                *   if in set, plot pixel
                */
               plot_logic (normcolor, alternate, count, max_iter,
                    scale, ix, iy, oset);
          }
     }
}




/*------------------------------*/
/*   plot_logic         */
/*------------------------------*/
plot_logic (normcolor, alternate, count, max_iter, scale, ix, iy, oset)
int  normcolor;
int  alternate;
int  count;
```

```
int  max_iter;
int  scale;
int  ix;
int  iy;
int  oset;
{

/*
 *   logic to figure if point is in set or not and what color to plot it
 */

     if (normcolor)
     {
          if (alternate)
          {
               if ((count % 2) && (count < max_iter))
                    do_pixel (scale, ix, iy, oset, BLACK);
               else
                    do_pixel (scale, ix, iy, oset, WHITE);
          }
          else if (count >= max_iter)
          {
               do_pixel (scale, ix, iy, oset, BLACK);
          }
          else
          {
               do_pixel (scale, ix, iy, oset, WHITE);
          }
     }
     else
     {
          if (alternate)
          {
               if (!((count % 2) && (count < max_iter)))
                    do_pixel (scale, ix, iy, oset, BLACK);
               else
                    do_pixel (scale, ix, iy, oset, WHITE);
          }
          else if (count < max_iter)
          {
               do_pixel (scale, ix, iy, oset, BLACK);
          }
          else
          {
               do_pixel (scale, ix, iy, oset, WHITE);
          }
     }
}
```

```
/*
 *   these functions are all required but these are atari-specific
 */

#ifdef atarist
#include <osbind.h>              /* atari ST (BIOS) */
#endif


/*------------------------------*/
/*   init          */
/*------------------------------*/
init ()
{

/*
 *   start up thingies (open windows, clear screens, etc)
 */

#ifdef atarist
     /*
      *   clear screen, hide cursor, init timer...
      */
     clr_screen ();
     cursor (0);
     do_time (0);
#endif

}




/*------------------------------*/
/*   finish             */
/*------------------------------*/
finish (hold_scrn)
int  hold_scrn;
{

/*
 *   finish up...
 */

#ifdef atarist
     if (hold_scrn)
```

```
     {
          wait_key ();
     }
     cursor (1);

     exit (0);
#endif

}




/*------------------------------*/
/*   check_key     */
/*------------------------------*/
check_key ()
{

/*
 *   ckecks for a key and flushes keyboard buffer.
 */

#ifdef atarist
     /*
      *   this is for atari ST using BIOS.
      */
     if (Bconstat (2))           /* if CONSOLE has a char... */
     {
          while (Bconstat (2))        /* read char while there are */
               Bconin (2);       /* chars to read (flush) */

          return (1);            /* yes, there was a key */
     }

     return (0);                 /* no key */

#endif /*atarist*/

}




/*------------------------------*/
/*   wait_key      */
/*------------------------------*/
wait_key ()
{

/*
 *   waits for a key and flushes keyboard buffer.
 */
```

```
#ifdef atarist
     /*
      *   this is for atari ST using BIOS.
      */
     while (!Bconstat (2))           /* wait for a key... */
          ;

     while (Bconstat (2))            /* keep reading while there */
          Bconin (2);            /* are key inputs... */

#endif /*atarist*/

}




/*------------------------------*/
/*   do_pixel      */
/*------------------------------*/
do_pixel (scale, ix, iy, oset, color)
int  scale;
int  ix;
int  iy;
int  oset;
int  color;
{

/*
 *   put a pixel or BIG pixel.
 */

#ifdef atarist
     /*
      *   this is for atari ST using line A.
      */
     extern long   linea0 ();
     extern int    linea1 ();

     static long   A_ptr = 0L;   /* to line-A struct */

     int      i,
              j;
     int      num;


     /*
      *   init line A if needed...
      */
     if (!A_ptr)
          A_ptr = linea0 ();
```

```
     /*
      *   if not fine scale, we need to plot BIG pixels
      */
     if (scale < FINE)
     {
          num = FINE / scale;

          for (i = 0; i < num; i++)
          {
               for (j = 0; j < num; j++)
               {
                    linea1 (A_ptr,     oset+(ix*num+i), oset+(iy*num+j), color);
               }
          }
     }
     else
     {
          linea1 (A_ptr, oset + ix, oset + iy, color);
     }

#endif /*atarist*/

}




/*------------------------------*/
/*   do_line            */
/*------------------------------*/
do_line (x1, y1, x2, y2)
int  x1, y1, x2, y2;
{

/*
 *   draw a line.
 */

#ifdef atarist
     /*
      *   this is for atari ST using line A.
      */
     extern long   linea0 ();
     extern int    linea3 ();

     static long   A_ptr = 0L;


     /*
      *   line A init first, if needed
      */
     if (!A_ptr)
```

```
          A_ptr = linea0 ();

     linea3 (A_ptr, x1, y1, x2, y2, 0, 0xFFFF, 0);

#endif /*atarist*/

}




/*------------------------------*/
/*   cursor             */
/*------------------------------*/
cursor (visible)
int  visible;
{

/*
 *   turn cursor on/off
 */

#ifdef atarist
     /*
      *   for atari ST. uses gemdos string output and vt52 escapes
      */
     if (visible)
          Cconws ("\33e");
     else
          Cconws ("\33f");

#endif /*atarist*/

}




/*------------------------------*/
/*   mv_cursor     */
/*------------------------------*/
mv_cursor (col, row)
int  col;
int  row;
{

/*
 *   move cursor to row,col (0,0 is UL corner). for printing
 */

#ifdef atarist
     /*
      *   for atari ST. uses vt52 escapes
```

```
     */
     printf ("\33Y%c%c", (char) (row + ' '), (char) (col + ' '));
     fflush (stdout);

#endif /*atarist*/

}




/*------------------------------*/
/*   clr_screen         */
/*------------------------------*/
clr_screen ()
{

/*
 *   clear screen, home cursor
 */

#ifdef atarist
     /*
      *   for atari ST. uses gemdos string output and vt52 escapes
      */
     Cconws ("\33E");

#endif /*atarist*/

}




/*------------------------------*/
/*   do_time            */
/*------------------------------*/
long do_time (opt)
int  opt;               /* 0=init,1=start,2=end (ret elapsed)*/
{

/*
 *   timing functions
 */

#ifdef atarist
     /*
      *   for atari ST. uses gemdos and sys variables (200 hz timer).
      */
     static unsigned long   start;
     static unsigned long   stop;

     long          savessp;
```

```
     unsigned long      elapsed;      /* seconds */


     switch (opt)
     {
     case 0:                         /* reset clock */
     case 1:                         /* start clock */
          savessp = Super (0L);
          start   = *(long *) (0x4ba);
          Super (savessp);
          break;

     case 2:                         /* stop and elapsed */
          savessp = Super (0L);
          stop    = *(long *) (0x4ba);
          Super (savessp);

          if (stop < start)
               elapsed = (long) (stop - start + 0x7FFFFFFFL) / 200L;
          else
               elapsed = (long) (stop - start) / 200L;

          return ((long) elapsed);
     }

     return (0L);

#endif /*atarist*/

}
22842!Funky!Stuff!
echo x - version.h
cat > version.h << '22842!Funky!Stuff!'
/*
 *   version.h for fract
 */

#define VERSION        100

char          version_G[] = "fract v1.0 (mono) wjr 10/89";
char          copyrite[]  = "(c) 1989 Bill Rosenkranz";

22842!Funky!Stuff!
echo x - linea0.s
cat > linea0.s << '22842!Funky!Stuff!'
*
*    NAME:         linea0 - return pointer to line A struct
*
*
*    SYNOPSIS: ptr = (struct line_a *) linea0 ();
*
*
*    AUTHOR:       (c) 1989 Bill Rosenkranz
```

```
*
*             net: rosenkra%boston@hall.cray.com
*             CIS: 71460,17
*             GENIE:    W.ROSENKRANZ
*
*             Feel free to copy. Leave this comment alone! No other
*             restrictions on copying or hacking. Use as you like.
*

Init=$A000

     .globl    _linea0

     .text

_linea0:
     link a6,#0              * stack frame
     movem.l   a0-a2,-(a7)        * save regs we clobber (except d0)

     dc.w Init

* pointer to struct now in d0 and a0. a1 -> array of font headers, a2 -> array
* of linea functions

     movem.l   (a7)+,a0-a2        * restore regs
     unlk a6             * fix stack frame

     rts                 * d0 is pointer

     .end

22842!Funky!Stuff!
echo x - linea1.s
cat > linea1.s << '22842!Funky!Stuff!'
*
*    NAME:         linea1 - put pixel
*
*
*    SYNOPSIS: void linea1 (struct line_a *ptr, int x, int y,
*                  int color);
*
*
*    AUTHOR:       (c) 1989 Bill Rosenkranz
*
*             net: rosenkra%boston@hall.cray.com
*             CIS: 71460,17
*             GENIE:    W.ROSENKRANZ
*
*             Feel free to copy. Leave this comment alone! No other
*             restrictions on copying or hacking. Use as you like.
*

* linea struct offsets:
```

```
INTIN=8
PTSIN=12

* args: (wrt stack frame ptr a6)

ptr=8
x=12
y=14
color=16


     .globl   _linea1

     .text

_linea1:
     link a6,#0
*    movem.l  a0,-(a7)

     move.l   ptr(a6),a0         * linea struct ptr
     move.l   INTIN(a0),a1
     move.l   PTSIN(a0),a2
     move.w   color(a6),(a1)          * put color in intin table
     move.l   x(a6),(a2)         * put coord in ptsin table

     dc.w $A001              * draw pixel

*    movem.l  (a7)+,a0
     unlk a6

     rts


     .end

22842!Funky!Stuff!
echo x - linea2.s
cat > linea2.s << '22842!Funky!Stuff!'
*
*    NAME:         linea2 - get pixel
*
*
*    SYNOPSIS: int linea2 (struct line_a *ptr, int x, int y);
*
*
*    AUTHOR:       (c) 1989 Bill Rosenkranz
*
*             net: rosenkra%boston@hall.cray.com
*             CIS: 71460,17
*             GENIE:   W.ROSENKRANZ
*
*             Feel free to copy. Leave this comment alone! No other
```

```
*             restrictions on copying or hacking. Use as you like.
*

* linea struct offsets:

PTSIN=12

* args: (wrt stack frame ptr a6)

ptr=8
x=12
y=14


     .globl   _linea2

     .text

_linea2:
     link a6,#0
     movem.l  a0-a1,-(a7)

     move.l   ptr(a6),a0         * linea struct ptr
     move.l   PTSIN(a0),a1
     move.l   x(a6),(a1)         * put coord in ptsin table

     dc.w $A002              * get pixel. returns d0.w

     movem.l  (a7)+,a0-a1
     unlk a6

     rts


     .end

22842!Funky!Stuff!
echo x - linea3.s
cat > linea3.s << '22842!Funky!Stuff!'
*
*    NAME:         linea3 - draw line from x1,y1 to x2,y2
*
*
*    SYNOPSIS: void linea3 (struct line_a *ptr, int x1, int y1,
*                  int x2, int y2, int mode, int mask, int opt);
*
*
*    AUTHOR:       (c) 1989 Bill Rosenkranz
*
*             net: rosenkra%boston@hall.cray.com
*             CIS: 71460,17
*             GENIE:    W.ROSENKRANZ
*
```

```
*             Feel free to copy. Leave this comment alone! No other
*             restrictions on copying or hacking. Use as you like.
*

* linea struct offsets:

CBIT0=24
CBIT1=26
CBIT2=28
CBIT3=30
LSTLIN=32
LNMASK=34
WMODE=36
X1=38
Y1=40
X2=42
Y2=44


* args:  (wrt stack frame ptr a6)

ptr=8
x_1=12
y_1=14
x_2=16
y_2=18
mode=20
mask=22
opt=24

     .globl   _linea3

     .text

_linea3:
     link a6,#0

     move.l   ptr(a6),a0         * line A init (a0 -> structure)

     move.w   #1,CBIT0(a0)       * plane 0 (all)
     move.w   #1,CBIT1(a0)       * plane 1 (color only)
     move.w   #1,CBIT2(a0)       * plane 2 (lo res only)
     move.w   #1,CBIT3(a0)       * plane 3 (lo res only)
     move.w   opt(a6),LSTLIN(a0) * draw last pixel of line?
     move.w   mask(a6),LNMASK(a0)    * line mask (solid=FFFF)
     move.w   mode(a6),WMODE(a0) * write mode (replace=0)
     move.l   x_1(a6),X1(a0)         * coord of points
     move.l   x_2(a6),X2(a0)

     dc.w $A003              * draw line

     unlk a6
```

```
     rts

     .end

22842!Funky!Stuff!


From -5088488069381600815
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,a22bb0503a5d559a,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-11-11 00:03:29 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!tut.cis.ohio-state.edu!
gem.mps.ohio-state.edu!brutus.cs.uiuc.edu!lll-winken!sun-barr!newstop!sun!
sunfedcomm!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i005: xxed -- Hex/ASCII file editor  part01/03
Keywords: shar, C
Message-ID: <34744@grapevine.uucp>
Date: 11 Nov 89 08:03:29 GMT
Sender: daemon@sunfedcomm
Reply-To: charltn@ccu.umanitoba.ca (Jim Charlton)
Lines: 1689
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: charltn@ccu.umanitoba.ca (Jim Charlton)
Posting-number: Volume 3, Issue 5
Archive-name: xxed/part01

[Documentation is included in the sharfile. -sg]

#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
# This is part 1 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
#
#    Run the following text with /bin/sh to create:
#      BUFMAN.C
#      BUTTON.C
#      CURSOR.C
#      EDIT.C
#      EVENTS.C
#      FILES.C
#      FNCTLST.DOC
#      GLOBALS.H
#      INIT.C
#      KEYS.C
#      MAIN.C
#      MAKEFILE
#      MENU.C
#      ONEPAGE.C
```

```
#      PRINTER.C
#      SEARCH.C
#      SEND.C
#      SLIDER.C
#      WIND.C
#      XXED.DEF
#      XXED.H
#      XXED.RSC
#      XXEDRSC.C
#      XXEDSRC.DOC
#
if test -r s2_seq_.tmp
then echo "Must unpack archives in sequence!"
     next=`cat s2_seq_.tmp`; echo "Please unpack part $next next"
     exit 1; fi
sed 's/^X//' << 'SHAR_EOF' > BUFMAN.C &&
X#include "stdio.h"
X#include "gemdefs.h"
X#include "obdefs.h"
X#include "osbind.h"
X#include "globals.h"
X#include "xxed.h"
X/*
XNote:  Each window has its own file buffer.  This file buffer is composed
Xof a linked list of blocks.  addmember() adds a new block to the end of the
Xlist of blocks in the buffer, or puts the first block into the list if
Xthere were no blocks in the list.  The size of the blocks in the buffer
Xis set with the globabl constant BLOCKSIZE.  Using a linked list of blocks
Xfor a buffer simplifies memory management and speeds up inserts and deletes
Xsince only the block containing the data in question need be modified.
Xinsert_member() inserts a block into the linked list and splits the data
Xin the previous block between it and the new block.  This leaves two half
Xfilled blocks for insertions.  dispose_member() deletes a block out of the
Xlist....  happens when all of the data in a block is deleted or cut.
Xdispose_buf() will delete the entire file buffer by deleting all of its
Xblocks
X*/
X
X/* add a member to the linked list at the end  */
X
Xlinkbufptr addmember(thewin)
X    windowptr thewin;
X{
X    linkbufptr newmem, amem;
X    int c;
X
X    /* allocate space for the new member */
X    newmem = (linkbufptr)malloc(sizeof(linkbuf));
X    if(newmem == NULL) return NULL;  /* no more memory available  */
X
X    /* initialize the newmem linkbuf */
X
X    newmem->next = NULL;
```

```
X    bzero(newmem->block,BLOCKSIZE);
X     newmem->inuse = 0;
X    /* insert into the linked list       */
X
X    if(!thewin->headptr)    /* if thewin->headptr is NULL   */
X      thewin->headptr = newmem;
X    else
X    {  amem = thewin->headptr;
X           while(amem->next)
X             amem = amem->next;
X       amem->next = newmem;
X    }
X      return(newmem);
X}
X
X/* insert a new linked buffer in the list for any list */
X
Xinsert_member(bufptr)
X    linkbufptr    bufptr;
X{
X    linkbufptr newmem;
X    int totransfer, left, i;
X
X    /* allocate space for the new member */
X    newmem = (linkbufptr)malloc(sizeof(linkbuf));
X    if ( newmem == NULL)  /* no more memory available  */
X        return(-1);
X    /* initialize the newmem linkbuf and insert into list */
X
X    newmem->next = bufptr->next;
X    bzero(newmem->block,BLOCKSIZE);
X     newmem->inuse = 0;
X    bufptr->next = newmem;
X
X    /* now split the contents of the block in bufptr between old & new */
X
X    totransfer = bufptr->inuse - bufptr->inuse/2;
X     left = bufptr->inuse/2;
X/*  strncpy(newmem->block,bufptr->block+left,totransfer);
     */
X    bcopy(bufptr->block+left,newmem->block,totransfer);
X     newmem->inuse = totransfer;
X     bufptr->inuse = left;
X}
X
Xdispose_buf(thewin)
X    windowptr     thewin;
X{
X    int button;
X    linkbufptr    amem;
X    char str[160];
X
X    if(thewin->changed == TRUE)
```

```
X    {    sprintf(str,"[0][ | File %s | has been changed. | Do you want to SAVE |
before closing? ][ YES | NO ]", thewin->title);
X         button = form_alert(1, str);
X         if(button == 1)
X              handle_file(SAVE);
X    }
X
X     for (amem=thewin->headptr;amem;amem = thewin->headptr)
X    {    dispose_member(thewin,amem);
X    }
X
X}
Xdispose_member(thewin,memtodel)
X    windowptr thewin;
X    linkbufptr memtodel;
X
X{   linkbufptr amem;
X    int button;
X
X    if(thewin->headptr)
X    {
X         if (thewin->headptr == memtodel)
X              thewin->headptr = thewin->headptr->next;
X         else
X       {  for (amem = thewin->headptr;(amem->next != memtodel) ;amem=amem->next)
X                       if (!amem) break;
X        /* search list 'til find memtodel in amem->next or amem==NULL */
X        if(amem)
X                amem->next = amem->next->next;
X        else
X              {    button = form_alert(1, "[1][ Internal Error: |  member to
delete not in list. ][OK]");
X              shutdown(2);
X              }
X       }
X         free((char *)memtodel); /* free up space of deleted member  */
X    }
X}
X
SHAR_EOF
chmod 0600 BUFMAN.C || echo "restore of BUFMAN.C fails"
sed 's/^X//' << 'SHAR_EOF' > BUTTON.C &&
X#include "xxed.h"
X#include <gemdefs.h>
X#include <obdefs.h>
X#include <osbind.h>
X
X#include "globals.h"
X
X
Xdo_button(mousex,mousey)
X    int  mousex,mousey;
X{
```

```
X    windowptr thewin;
X    int  x,y,x2,y2,status,lastw,lasth;
X    long oldpos, newpos;
X
X
X/* WARNING  There are multiple 'returns' in this function  */
X
X    thewin = thefrontwin;
X    status = 1;
X    lasth=lastw=0;
X    oldpos = thewin->position;
X    while(status==1)
X    {    vq_mouse(thewin->graf.handle,&status,&x,&y);
X         if ((abs(x-mousex))>6 || ((abs(y-mousey))>6))
X              graf_rubberbox(mousex,mousey,6,6,&lastw,&lasth);
X    }
X    if(lastw>7 || lasth>7)
X    {
X
X         newpos = calc_pos(mousex,mousey);
X         if(newpos == -1L) return;
X         if(newpos == -2L)
X         {    thewin->position = thewin->flen-1;
X              return;
X         }
X         thewin->startmark = newpos;
X         newpos = calc_pos(x-3,y-3);
X         if(newpos == -1) return;
X         thewin->endmark = newpos<thewin->flen-1 ?
X                                 newpos : thewin->flen-2;
X         thewin->position = thewin->endmark;
X         if(thewin->startmark <= thewin->endmark) /* it must !! */
X         {    send_redraw(thewin);
X              thewin->markson = TRUE;
X              menubar[CUT].ob_state = NORMAL;
X              menubar[COPY].ob_state = NORMAL;
X              menubar[ERASE].ob_state = NORMAL;
X         }
X    }
X    else
X    {    newpos = calc_pos(mousex,mousey);
X         if(newpos == -1) return;
X         if(newpos == -2L)
X         {    thewin->position = thewin->flen-1;
X              return;
X         }
X    }
X    thewin->position = newpos;
X}
X
X
Xlong calc_pos(mousex,mousey)
X    int  mousex,mousey;
```

```
X{
X    int x,y,w,h,i;
X    windowptr thewin;
X    long pos;
X    int button, xin;
X
X         thewin = thefrontwin;  /* move cursor around in frontwin */
X         x = thewin->work.g_x;
X         y = thewin->work.g_y;
X         w = thewin->work.g_w;
X         h = thewin->work.g_h;
X         mousex -= 5;
X
X         if(mousex > x + w) mousex = x + w - gl_wchar;
X         if(mousex > (x + gl_wchar * 48) && mousex < (x + gl_wchar * 64))
X              {    xin = (mousex - x - gl_wchar*48)/gl_wchar;
X                   mousex = x + xoffset[xin]*gl_wchar+1;
X                   inhex = FALSE;
X                   wr_ins_rpl(ins);
X                        /* go to ascii input   */
X              }
X         else
X              {    inhex = TRUE;
X                   wr_ins_rpl(ins);       /* hex input  */
X              }
X         if(mousex < x) mousex = x + gl_wchar/2;
X         if(mousex > (x + gl_wchar * 48)) mousex = x + gl_wchar * 48;
X         if(mousey < y) mousey = y + gl_wchar/2;
X         if(mousey > y + h) mousey = y + h - gl_wchar/2;
X              {    for (i=0;xoffset[i]*gl_wchar < (mousex-x);i++);
X                   pos = 16*((mousey-y)/gl_hchar) +
X                                       i + thewin->topchar - 1;
X              }
X
X    if(pos>thewin->flen-1)
X         {
X    return(-2L); /* trying to move cursor outside file !   */

X         }
X    if(thewin->icount!=0) /* don't allow cursor move til finish entry*/
X         {
X    Bconout(2,7);
X    button = form_alert(1, "[1][ | | Enter a HEX digit or Backspace ][ OK ]");

X    return(-1L);
X         }
X              pos = pos < (thewin->flen-1) ? pos : (thewin->flen-1) ;
X
X              return pos;
X}
X
X
SHAR_EOF
```

```
chmod 0600 BUTTON.C || echo "restore of BUTTON.C fails"
sed 's/^X//' << 'SHAR_EOF' > CURSOR.C &&
X#include <osbind.h>
X#include <gemdefs.h>
X#include <obdefs.h>
X
X#include "globals.h"
X
X
Xputcur(thewin,type)
X    windowptr thewin;
X    int           type;  /* actually, type is never used */
X
X{   int           xpos, ypos;
X    int diff, num, rect[4];
X    int  x,y,w,h;
X    int  xacur, yacur;
X    long winlines, bottomchar;
X
X    winlines = thewin->work.g_h/gl_hchar; /* no of lines can put in window  */
X    bottomchar = thewin->topchar + (winlines * 16) - 1;
X
X    if( (thewin->position < thewin->topchar) ||
X                        (thewin->position > bottomchar) )
X         return;  /* don't try to draw cursor unless on page  */
X
X/* set the clipping rectange in case it was not set    */
X
X    wind_get(thewin->handle, WF_WORKXYWH, &x, &y, &w, &h);
X    rect[0] = x;
X    rect[1] = y;
X    rect[2] = x+w-1;
X    rect[3] = y+h-1;
X
X    vs_clip(thewin->graf.handle,1,rect);
X
X    if (thewin = thefrontwin)  /*  cursor action for frontwin only */
X  {
X         diff = thewin->position - thewin->topchar;
X         num = diff % 16;
X         thewin->ycur = (diff/16)*gl_hchar + thewin->work.g_y + 1;
X         thewin->xcur = xoffset[num]*gl_wchar + gl_wchar -1
X                                                    + thewin->work.g_x;
X         yacur = thewin->ycur;
X         xacur = thewin->work.g_x + 49*gl_wchar + num*gl_wchar;
X
X    rect[0] = thewin->xcur+1;
X    rect[1] = thewin->ycur;
X    rect[2] = rect[0]+gl_wchar*2-1;
X    rect[3] = rect[1]+gl_hchar-1;
X
X    vsf_perimeter(thewin->graf.handle,0);
X    vsf_interior(thewin->graf.handle,1);
```

```
X    vsf_style(thewin->graf.handle,1);
X    vswr_mode(thewin->graf.handle,3);
X    v_bar(thewin->graf.handle,rect);
X
X    rect[0] = xacur -1;
X    rect[1] = yacur;
X    rect[2] = rect[0] + gl_wchar-1;
X    rect[3] = rect[1] + gl_hchar-1;
X    v_bar(thewin->graf.handle,rect);
X
X    vswr_mode(thewin->graf.handle,1);
X  }
X}
X
SHAR_EOF
chmod 0600 CURSOR.C || echo "restore of CURSOR.C fails"
sed 's/^X//' << 'SHAR_EOF' > EDIT.C &&
X#include <gemdefs.h>
X#include <obdefs.h>
X#include <osbind.h>
X#include <stdio.h>
X#include "xxed.h"
X
X#include "globals.h"
X
X
Xdelete_one(thewin,pos,flag) /* delete one byte */
X    windowptr thewin;
X    long     pos;
X    int  flag;
X{
X        linkbufptr amem;
X        char *addr, *addr2;
X        int  button;
X
X/*  flag is 0 when saving the file in which case can delete last byte */
X    if( (thewin->position==thewin->flen-1) && flag) /* last byte */
X    {    button = form_alert(1, "[1][ The last byte in the file is | a dummy byte
and cannot be | changed or deleted. It is | not saved with the file. ][ OK ]");

X        return;
X    }
X
X        amem = thewin->headptr;
X        while (amem->inuse<=pos)
X             {    pos = pos - amem->inuse;
X                  amem = amem->next;
X             }
X        addr = (char *)(amem->block+pos);
X    for (addr2=addr;addr2<(char *)amem->block+amem->inuse;addr2++)
X        *addr2 = *(addr2+1);
X
X        amem->inuse--;
```

```
X         thewin->flen--;
X    if(thewin->position>=thewin->flen)
X         thewin->position= thewin->flen-1;
X    if(flag) /* don't set to partial if deleting last byte in file
     */
X         thewin->changed = TRUE;
X}
X
Xcutit(thewin) /* cut out the marked text   */
X    windowptr thewin;
X{
X    char      *addr;
X    linkbufptr    amem;
X    long      pos, togo;
X    int           inthis, dummy;
X
X    pos = thewin->startmark;
X        if (pos<=thewin->flen)
X    {    amem = thewin->headptr;
X         while (amem->inuse <= pos)
X              {    pos = pos - amem->inuse;
X                   amem = amem->next;
X              }
X         inthis = amem->inuse - pos;
X    }
X    togo = thewin->endmark - thewin->startmark + 1;
X    if(togo > 0)
X    {    while (inthis < togo)
X         {    amem->inuse = pos;
X              if(amem->inuse==0)
X                   dispose_member(thewin,amem);
X              amem = amem->next;
X              togo -= inthis;
X              thewin->flen -= inthis;
X              inthis = amem->inuse;
X              pos = 0;
X         }
X         if(togo > 0)
X         {    bcopy(amem->block+pos+togo,amem->block+pos,
X                                       (int)(amem->inuse-togo-pos));
X              amem->inuse -= togo;
X              thewin->flen -= togo;
X         }
X    }
X    thewin->position = thewin->startmark;
X    if(thewin->topchar >= thewin->position)
X         thewin->topchar = thewin->position&~0x7;
X}
X
X
Xcopy(thewin) /* copy the marked text to the cutbuffer */
X    windowptr thewin;
X{
```

```
X    linkbufptr    amem;
X    long     pos;
X    long     tocopy;
X    int           bufbytes, num, button;
X    char     *addr, *addr2;
X
X    pos = thewin->startmark;
X
X        if (pos<=thewin->flen)
X    {    amem = thewin->headptr;
X         while (amem->inuse <= pos)
X              {    pos = pos - amem->inuse;
X                   amem = amem->next;
X              }
X         addr = (char *)(amem->block+pos);
X    }
X
X
X    if(cutbuffer) free(cutbuffer);
X    cutlength = thewin->endmark - thewin->startmark +1;
X    tocopy = cutlength;
X    if (cutlength<=0) return; /* bail out if end is before start */
X    cutbuffer = malloc((unsigned)(tocopy+2));
X
X    if (cutbuffer == NULL)
X    {    button = form_alert(1, "[1][ Sorry! | Out of memory creating cut
buffer. ][OK]");
X         return;
X    }
X
X    bufbytes = amem->inuse - pos;
X    addr2 = cutbuffer;
X
X    while(tocopy > 0)
X         {    num = (tocopy < bufbytes) ? tocopy : bufbytes;
X              bcopy(addr,addr2,num);
X              tocopy -= bufbytes;
X         if(tocopy > 0)
X              {    amem = amem->next;
X                   addr = amem->block;
X                   bufbytes = amem->inuse;
X                   addr2 += num;
X              }
X         }
X    *(cutbuffer + cutlength ) = 0;
X}
X
Xpaste(thewin) /* paste the cutbuffer into the file before the cursor */
X    windowptr thewin;
X{
X    insert_it(thewin,cutlength,cutbuffer);
X    send_vslid(thewin);
X    send_redraw(thewin);
```

```
X}
X
X/* insert_it() inserts incopy bytes from string pointed to by addr2
X   into the file before the position of the cursor
X*/
X
Xinsert_it(thewin,incopy,addr2)
X    windowptr thewin;
X    long incopy;
X    char *addr2;
X{
X    long     pos, insertpos, tocopy;
X    linkbufptr    amem;
X    char     *posaddr,*addr,*addr3,*addr4;
X    int           havenow, num, tomoveup, button, ret;
X
X    insertpos = thewin->position;
X    tocopy = incopy;
X    while(tocopy>0)
X    {
X         /* calculate amem that cursor is in */
X         pos = insertpos;
X              {    amem = thewin->headptr;
X                   while (amem->inuse < pos)
X                   {    pos = pos - amem->inuse;
X                        amem = amem->next;
X                   }
X              posaddr = (char *)(amem->block+pos);
X              }
X         if (amem->inuse == BLOCKSIZE)
X         {
X              ret = insert_member(amem);  /* free up some space for the insert */
X         if (ret == -1)  /* no more memory available  */
X    {    button = form_alert(1, "[1][ Out of memory. | Partial insert. ][OK]");
X              break;
X    }    /* recalculate pos and amem */
X              pos = insertpos;
X              {    amem = thewin->headptr;
X                   while (amem->inuse <= pos)
X                   {    pos = pos - amem->inuse;
X                        amem = amem->next;
X                   }
X              posaddr = (char *)(amem->block+pos);
X              }
X         }
X    /* pos is now position of cursor in number of bytes from amem->block */
X
X         addr = posaddr;
X         havenow = BLOCKSIZE - amem->inuse;
X         num = (tocopy < havenow) ? tocopy : havenow;
X         for(tomoveup = amem->inuse - pos-1;tomoveup>-2;tomoveup--)
X              *(addr+tomoveup+num) = *(addr+tomoveup);
X         bcopy(addr2,addr,num);
```

```
X         amem->inuse += num;
X         addr2 += num;
X         insertpos += num;
X         tocopy -= num;
X    }
X    thewin->flen += incopy;
X}
X
SHAR_EOF
chmod 0600 EDIT.C || echo "restore of EDIT.C fails"
sed 's/^X//' << 'SHAR_EOF' > EVENTS.C &&
X#include <gemdefs.h>
X#include <obdefs.h>
X#include <osbind.h>
X
X#include "globals.h"
X
X/*
X    Handle Application Events.
X*/
X
XTaskMaster()
X{
X    int event;              /* The event code.
          */
X    int button = TRUE; /* desired Button state
     */
X    int  message[8];        /* Event message buffer.
          */
X    int  mousex, mousey;    /* The current mouse position.
          */
X    int mousebutton;   /* The state of the mouse button    */
X
X    int keycode;       /*  The code for the key pressed.
     */
X    int keymods;       /*  The state of the keyboard modifiers.
X                               (shift, ctrl, etc). */
X    int clicks;             /*   The number of mouse clicks that occurred in the
X                               given time. */
X    windowptr thewin;
X    int  c;
X
X             sprintf(fsize_str," File size:%ld",0L);
X             sprintf(pos_str," Position:%ld",0L);
X    do {
X         if(thewin = thefrontwin)
X         {    graf_mouse(M_OFF, 0L);
X              putcur(thewin);    /* cursor on  */
X              graf_mouse(M_ON, 0L);
X         }
X         wr_ins_rpl(ins);
X
X         event = evnt_multi(
```

```
X             MU_MESAG | MU_BUTTON | MU_KEYBD,   /* set messages to respond to. */
X                       /* MU_MESAG    is a message event    */
X             1,                /* Time frame for events.
                        */
X             1,                /* Keyboard Event mask.
                        */
X             button,           /* desired key state
                        */
X             0, 0, 0, 0, 0,    /* rectangle one information (ignored)
         */
X             0, 0, 0, 0, 0,    /* rectangle two information (ignored)
         */
X             message,      /* The message buffer
                  */
X             0, 0,             /* Number of Ticks for Timer event.
         */
X             &mousex,      /* The x-coordinate of the mouse at event.  */
X             &mousey,      /* The y-coordinate of the mouse at event.  */
X             &mousebutton, /* The state of the mouse buttons at event. */
X             &keymods,     /* The state of the keyboard modifiers.     */
X             &keycode,     /* The key code for the key pressed.        */
X             &clicks           /* The number of times the event occurred
     */
X         );
X
X         if(thewin = thefrontwin)
X         {    graf_mouse(M_OFF, 0L);
X              putcur(thewin);    /* cursor off */
X              graf_mouse(M_ON, 0L);
X         }
X
X
X         if (event & MU_KEYBD)
X                   {
X                        if (thewin = thefrontwin)
X/*  if repos is true.  eg. cursor is off the page, redraw page */
X                             if (repos(thewin))
X                             {    partial = FALSE;
X                                  one_page(thewin,1);
X                                  redraw_vslider(thewin->handle);
X                             }
X                        draw_flag = FALSE;
X                        arro_flag = FALSE;
X                        slid_flag = FALSE;
X                        do_kbd(keycode, keymods);
X                   }
X         else if (event & MU_MESAG) {
X              switch (message[0]) {
X                   /*
X                        Window Support
X                   */
X                   case WM_REDRAW:
X                   case WM_TOPPED:
```

```
X                  case WM_FULLED:
X                  case WM_ARROWED:
X                  case WM_HSLID:
X                  case WM_VSLID:
X                  case WM_SIZED:
X                  case WM_MOVED:
X                  case WM_NEWTOP:
X                  case WM_CLOSED:
X                       do_window(message);
X                       break;
X
X                  /*
X                       Menu Support
X                  */
X                  case MN_SELECTED:
X                       do_menu(message);
X                       button ^= TRUE;
X                  break;
X
X                  /*
X                       Desk Accessory Support
X                  */
X                  case AC_OPEN:
X                  case AC_CLOSE:
X                  break;
X             }
X        }
X
X        else if (event & MU_BUTTON)
X        {
X                  if(button)
X                       do_button(mousex, mousey);
X                  button ^= TRUE;
X        }
X        if(thewin = thefrontwin)
X        {    sprintf(fsize_str," File size:%ld",thewin->flen-1);
X             sprintf(pos_str," Position:%ld",thewin->position);
X        }
X
X    }
X    while(1);
X}
SHAR_EOF
chmod 0600 EVENTS.C || echo "restore of EVENTS.C fails"
sed 's/^X//' << 'SHAR_EOF' > FILES.C &&
X
X#include "gemdefs.h"
X#include "obdefs.h"
X#include "osbind.h"
X#include "fcntl.h"
X#include "xxed.h"
X#include "globals.h"
X
```

```
X    int               drarray[16] = {AA,BB,CC,DD,EE,FF,GG,HH,II,JJ,
X                                          KK,LL,MM,NN,OO,PP};
X/*  The drive letters are defined in HEXED.H    */
X
Xread_file(thewin)
X    windowptr thewin;
X
X{   int fildes, c, button, ret;
X    long skres, flenr, toread, readnow, red;
X    linkbufptr    bufptr;
X    char str2[30];
X
X         graf_mouse(2,0L);
X           ret = 1;
X          fildes = open(thewin->title,O_BINARY);
X          skres = lseek(fildes,0L,0);       /* rewind */
X          flenr = Fseek(0L,fildes,2);        /* get file length */
X              if (flenr >= 1 && fildes != -1 && skres != -1)
X             {    skres = lseek(fildes,0L,0);       /* rewind again */
X             for(toread=flenr;toread>0;toread=toread-BLOCKSIZE)
X                  {    readnow = (toread < BLOCKSIZE) ? toread : BLOCKSIZE;
X                       bufptr = addmember(thewin);
X         if (bufptr == NULL)  /* no more memory available  */
X    {    button = form_alert(1, "[1][ Out of memory. | Partial file read. ][OK]");
X              ret = 0; break;
X    }
X                             /* read file to ram */
X                  if( (red = Fread(fildes,readnow,bufptr->block)) > 0 )
X                       {
X                             bufptr->inuse = red;
X                             thewin->flen += red;
X                       }
X                  else
X                       {    button = form_alert(1, "[1][ File read error ][OK]");
X                             ret = 0; break;
X                       }
X                   }
X             c = close(fildes);
X             }
X              else    {button = form_alert(1, "[1][ File read error | or file not
found ][OK]");
X                             ret = 0;
X                       }
X
X                  /* put extra char 00 at end of file  */
X         if(thewin->flen > 0)
X         {    if(bufptr->inuse < BLOCKSIZE)
X              {    bufptr->inuse += 1;
X                   thewin->flen += 1;
X              }
X              else
X              {    bufptr = addmember(thewin);
X              if (bufptr == NULL)  /* no more memory available  */
```

```
X                   {    button = form_alert(1, "[1][ Out of memory. | Partial file
read. ][OK]");
X                              ret = 0;
X                   }
X              else
X                   {    bufptr->inuse += 1;
X                        thewin->flen += 1;
X                   }
X              }
X         }
X         graf_mouse(0,0L);
X           return(ret);
X}
X
X
Xwrite_file(thewin)
X  windowptr thewin;
X{
X         int button,w;
X
X         drw_dialxy(dialog5,230,38/(16/gl_hchar));  /* may not work on colour   */
X         if (getfile(thewin))
X              save_file(thewin);
X         form_dial(FMD_FINISH, 0, 0, 0, 0, 230, 38/(16/gl_hchar),
X                                 dialog5->ob_width, dialog5->ob_height);
X
X}
X
Xsave_file(thewin)
X    windowptr thewin;
X{
X    int fildes, c, button;
X    long flenw;   /* length of buffer written to file */
X    linkbufptr    amem;
X    unsigned dummy;
X
X
X         if (thewin->flen < 2)
X              {button = form_alert(1, "[1][ Null file........ | Save aborted. ]
[OK]");
X                               return;
X              }
X
X         graf_mouse(2,0L);
X/* delete the dummy byte at end of file if it is still a 0x00 */
X
X         if((dummy = getbyte(thewin,thewin->flen-1))==0)
X/*  the 0 flag in the delete_one call allows for delete last byte
     */
X              delete_one(thewin,thewin->flen-1,0);
X
X                   fildes = open(thewin->title,O_RDONLY);
X              if (fildes > 0)
```

```
X             {    button =  form_alert(1, "[1][ File already exists. | Ok to
overwrite? ][OK|CANCEL]");
X                  if (button == 2)
X                       {    c = close(fildes);
X                            return;
X                       }
X                  else
X                       c = close(fildes);
X             }
X                  Fdelete(thewin->title);
X                  fildes = creat(thewin->title,O_RDWR);
X             if(fildes != -1)
X             {    for(amem=thewin->headptr;amem;amem=amem->next)
X                  {  if(flenw = Fwrite(fildes,amem->inuse,amem->block))
X                            ;
X                     else {button = form_alert(1, "[1][ File write error ][OK]");
X                             break;
X                            }
X                  }    c = close(fildes);
X                       thewin->changed = FALSE;
X             }
X             else
X             button = form_alert(1, "[1][ Error creating file | Try 'SAVE AS' ]
[OK]");
X         graf_mouse(0,0L);
X}
X
Xgetfile(thewin)  /* returns 1 with name in thewin->title, 0 with cancel */
X    windowptr thewin;
X
X{        int  fs_button, c, i, len;
X
X        c = fsel_input(fs_inpath,fs_insel,&fs_button);
X        if(( c!=0 ) && (fs_button != 0))
X        {
X             len = strlen(fs_inpath);
X             for(i=len;i>0;i--)  /* find last occurrence of \  */
X                  if(fs_inpath[i]=='\\') break;
X
X             strncpy(thewin->title,fs_inpath,i+1); /* copy pathname to title */
X             thewin->title[i+1] = '\0';
X             strcat(thewin->title,fs_insel);
X             wind_set(thewin->handle, WF_NAME, thewin -> title, 0, 0);
X             return(1);
X        }
X        else return(0);
X}
X
Xdelfile()  /* returns 1 with del, 0 with cancel */
X
X     {  int  fs_button, c, i, len;
X        char filespec[40];
X
```

```
X        drw_dialxy(dialog6,230,38/(16/gl_hchar));  /* may not work on colour   */
X                                        /* see form_dial at end of func */
X        c = fsel_input(fs_inpath,fs_insel,&fs_button);
X        form_dial(FMD_FINISH, 0, 0, 0, 0, 230, 38/(16/gl_hchar),
X                                 dialog6->ob_width, dialog6->ob_height);
X        if(( c!=0 ) && (fs_button != 0))
X        {
X             len = strlen(fs_inpath);
X             for(i=len;i>0;i--)  /* find last occurrence of \  */
X                  if(fs_inpath[i]=='\\') break;
X
X             strncpy(filespec,fs_inpath,i+1); /* copy pathname to filespec*/
X             filespec[i+1] = '\0';
X             strcat(filespec,fs_insel);
X             c = Fdelete(filespec);
X            if(c>=0)
X                  return(1);
X             else
X                  form_error(c);
X        }
X        else return(0);
X}
X
Xdrives()
X{
X    unsigned long drvs;
X    int               dr, cx, cy, cw, ch, button, idr;
X    int               dx, dy, dw, dh;
X    disk_info     myinfo;
X
X    dr_str[0] = 0;
X    for (dr = 1; dr < 17; dr++)
X              dialog4[drarray[dr]].ob_state = DISABLED;
X
X    idr = 0;
X    drvs = Drvmap();
X    for (dr = 1; dr < 17; dr++, drvs >>= 1)
X         if(drvs & 0x001)
X              dialog4[drarray[dr-1]].ob_state = NORMAL;
X
X         form_center(dialog4, &cx, &cy, &cw, &ch);
X         form_dial(FMD_START, 0, 0, 0, 0, cx, cy, cw, ch);
X         objc_draw(dialog4, 0, 10, cx, cy, cw, ch);
X    while((button = form_do(dialog4, 0)) != DREXIT)
X    {    for(dr = 1;dr<18;dr++)
X              if(drarray[dr-1] == button)
X                   idr = dr;
X         Dfree(&myinfo,idr);
X         sprintf(dr_str,"%ld Free bytes in disk %c",
X                   myinfo.b_free*myinfo.b_clsiz*myinfo.b_secsiz,
X                   'A' + (idr-1));
X         dialog4[button].ob_state = NORMAL;
X         objc_draw(dialog4, button, 10, cx, cy, cw, ch);
```

```
X         objc_draw(dialog4, DRBOX, 1, cx, cy, cw, ch);
X    }
X         form_dial(FMD_FINISH, 0, 0, 0, 0, cx, cy, cw, ch);
X         dialog4[DREXIT].ob_state = NORMAL;
X
X}
X
SHAR_EOF
chmod 0600 FILES.C || echo "restore of FILES.C fails"
sed 's/^X//' << 'SHAR_EOF' > FNCTLST.DOC &&
XMAIN.C
X    main() - executes initialization code and the starts the program.
X    set_table()      makes a table of ASCII hex bytes for 0 to 255
XINIT.C
X    init_resources() -     loads resource array
X    init_menu() -          sets up menu bar
X    init_dialog() -        sets up dialog boxes
X    init_path() -        sets up fs_inpath for fsel_input in files.c
X    shutdown(code)
X    cleanup() -           cleans up memory on shutdown
X    int open_vwork(form) - open virtual work area for each window
XEVENTS.C
X    TaskMaster() -         handles event_multi call
XWIND.C
X    do_window(message) -   handles all window messages
X    do_resize(message)
X    do_fullsize(handle)
X    do_update(message)
X    update_window(windhandle)
X    setclip(thewindow, r1)
X    windowptr findwindowptr(handle)
X    windowptr new_window(thekind)
X    open_window(thewin)
X    dispose_window(thewin)
X    dispose_win_resources(thewin)
X    make_frontwin(thewin)
X    rot_wind()
X    wind_blank(thewin)
XMENU.C
X    do_menu(message) -     handles menu messages
X    handle_desk(itemid)
X    handle_file(itemid)
X    new()  /* opens a new window with associated file  */
X    handle_marks(itemid)
X    start_mark(thewin)
X    end_mark(thewin)
X    clear_marks(thewin)
X    handle_edit(itemid)
X    handle_search(itemid)
XFILES.C
X    read_file(thewin) -    read file into ram buffer
X    write_file(thewin) -   write to a named file
X    save_file(thewin) -         write to current file thewin->title
```

```
X    getfile(thewin) - ret 1 name in thewin->title or 0 with cancel
X    delfile() - returns 1 with del, 0 with cancel
X    drives() -  display disk space on selected drive
XONEPAGE.C
X    one_page(thewin, blank) - draw one window full of data
X    unsigned getbyte(thewin,pos) - get any byte in file in ram
X    putbyte(thewin,pos,lnum) - put byte lnum into file at pos
X    one_line0(thewin,pos) - draw a single line of data in window
X    one_line2(thewin,pos)  /* handles mixed marked and unmarked */
X    one_line1(thewin,pos)  /* handles unmarked text    */
XBUFFMAN.C
X    linkbufptr addmember(thewin) - add new block to linked list
X    insert_member(bufptr) -  insert a new block into a lined list
X    dispose_buf(thewin) - delete an entire file buffer, all blocks in list
X    dispose_member(thewin,memtodel) - delete one block from linked list
XSLIDER.C
X    vhandler(message)
X    redraw_vslider(wihandle)    /* redraws vslider in new postion  */
X    size_vslider(wihandle)  /* set slider size  and redraws it */
X    scroll(thewin,upflag,dnflag) - scroll text in window
XCURSOR.C
X    putcur(thewin,type) - draws and undraws the cursor
XKEYS.C
X    do_kbd(keycode,keymods) - handles all keyboard input
X    do_kbd2(thewin,keycode,keymods) - handles hexadecimal input
X    cur_up(thewin)
X    cur_down(thewin)
X    cur_right(thewin)
X    cur_left(thewin)
X    check_scroll(thewin) - cursor off screen?  scroll or reposition
X    repos(thewin)  /* reposition and redraw with cursor on screen */
X    jump_pos(thewin) /* reset display parameters so cursor on screen */
X    ins_rpl(thewin)  /* enters hex charactes into the file */
X    enter_it(thewin,inbuf,lnum)- insert or replace byte redrawing screen
X    hex_chk(thewin, ch) /* checks to see if a hex character was entered */
X    wr_ins_rpl(insflag) /* writes insert/replace/hex/ascii on menu line */
X    cursor(x,y)   /* position the cursor*/
XBUTTON.C
X    do_button(mousex,mousey) - handles all mouse stuff including drag box
X    long calc_pos(mousex,mousey) claculates new position in file
XEDIT.C
X    delete_one(thewin,pos,flag) /* delete one byte */
X    cutit(thewin) /* cut out the marked text   */
X    copy(thewin) /* copy the marked text to the cutbuffer */
X    paste(thewin) /* paste the cutbuffer into the file before the cursor */
X    insert_it(thewin,incopy,addr2)
X         inserts incopy bytes from string pointed to by addr2
X         into the file before the position of the cursor
XSEARCH.C
X    find(thewin) /* handles search and replace dialog box  */
X    char *chk_sstr_len(thewin) /* check search string length */
X    char *chk_rstr_len(thewin) /* check replace string length */
X    find0(thewin) /* handles results from srch/rpl dialog */
```

```
X    forward1(thewin,sstr) /* searches forward */
X    long forward2(thewin,sstr)
X    rp_forw(thewin,sstr) /* replace going forward */
X    back1(thewin,sstr) /* search backwards */
X    long back2(thewin,sstr)
X    ronce(thewin,sstr,rstr) /* replace once */
X    rall(thewin,sstr,rstr) /* replace all */
X    rver(thewin,sstr,rstr) /* replace with verify */
X    do_dialog(dialog)   /* draw dialog boxes centered with form_do  */
X    do_dialxy(dialog,x,y) /* draw dialog at specific position with form_do */
X    drw_dialxy(dialog,x,y) /* draw object at a position.. no form_do */
XSEND.C
X    send_vslid(thewin) /* send application slider message */
X    send_redraw(thewin) /* send application a redraw message */
X    immed_redraw(thewin) /* bypass event multi and do a screen redraw */
X    send_arrow(thewin,direction) /* send application an arrow message */
XPRINTER.C
X    print(thewin) /* send file or block to printer */
X    print_line(thewin,pos) /* send a line to the printer */
X    pr_print(str) /* send a string to the printer */
X
X
X
X
X
X
X
SHAR_EOF
chmod 0600 FNCTLST.DOC || echo "restore of FNCTLST.DOC fails"
sed 's/^X//' << 'SHAR_EOF' > GLOBALS.H &&
X
X#ifndef TRUE
X#define TRUE      1
X#define FALSE          0
X#define NULL      0L
X#endif
X
X#define WORKSIZE  1
X#define BORDSIZE  0
X
X#define  min(a, b) ((a < b)? a : b)
X#define max(a, b) ((a > b)? a : b)
X
X/*
X    Each window will have a grafport so that it is easy to maintain
X    drawing environments accross windows.
X*/
Xtypedef struct _grafport {
X    int      handle;       /*  Virtual workstation handle
                        */
X    MFDB mfdb;         /*  memory definition block for virtual workstation
     */
X} grafport;
```

```
X
X
X/*
X    Window definition.
X*/
Xtypedef struct _windowrec {
X    struct _windowrec  *next;        /*  Window link
                        */
X    int                     handle;       /* Window Handle
                        */
X    int                     kind;         /* The windows type
                        */
X    int                     fullsize;/* In full-screen mode
                   */
X    int                     vslidepos;    /* vertical slider position
               */
X    char               title[30];    /* Title of window
                   */
X    GRECT                   box;      /* Window Box dimensions
               */
X    GRECT                   work;         /* Window content region dimensions
     */
X    grafport           graf;         /* Window graphics port definition  */
X    int                     (*updateproc)(); /* Pointer to update procedure */
X/*********************************************************************/
X/* the following are specific to the program HEXED                   */
X
X    struct _linkbuf        *headptr;
X    long               startmark;
X    long               endmark;
X    int                     markson;
X    int                xcur;         /* xpos of cursor on screen
               */
X    int                     ycur;       /* ypos of cursor on screen
               */
X    long               topchar;  /* number of char at top left
          */
X    long               flen;         /* length of file in total
               */
X    long               position;/* position of cursor in file
          */
X    char               input[2];/* input buffer
                   */
X    int                     icount;       /* input buffer end pointer
               */
X    int                     changed;
X/*********************************************************************/

X} windowrec;
Xtypedef windowrec *windowptr;
X
X
X/*
```

```
X    Declare the GEM specific variables required by the ROM routines.
X*/
X
Xextern int contrl[12];
Xextern int intin[256],  ptsin[256];
Xextern int intout[256], ptsout[256];
Xextern gl_apid;
X
X
X/*
X    Application Global Variables.
X*/
Xextern OBJECT         *menubar;
Xextern OBJECT         *dialog2;
Xextern OBJECT         *dialog1;
Xextern OBJECT         *dialog3;
Xextern OBJECT         *dialog4;
Xextern OBJECT         *dialog5;
Xextern OBJECT         *dialog6;
Xextern int            phys_handle;
Xextern char           table[600];
Xextern int            gl_hchar;
Xextern int            gl_wchar;
X
X#ifdef extern
Xwindowptr    firstwindow = NULL;       /* in MAIN.C these lines used  */
Xwindowptr    thefrontwin = NULL;
X#else
Xextern windowptr  firstwindow;
Xextern windowptr  thefrontwin;
X#endif
X
X/*
X    Functions
X*/
Xextern char       *malloc();
Xextern windowptr  new_window();
Xextern windowptr  frontwindow();
Xextern windowptr  findwindowptr();
X
X/* globals for HEXED      */
X
X#define BLOCKSIZE 4096
Xtypedef struct  _linkbuf {
X    struct  _linkbuf   *next;
X    char               block[BLOCKSIZE];
X    long               inuse;
X} linkbuf;
Xtypedef linkbuf *linkbufptr;
X
Xextern linkbufptr addmember();
Xextern linkbufptr add_cutmem();
Xextern int read_file();
```

```
Xextern long calc_pos();
X
X#ifdef extern
X    char fsize_str[32]={" File size:"};
X    char pos_str[32]={" Position:"};
X    char s_str[32];
X    char r_str[32];
X    char dr_str[32]={""};
X    int  xoffset[33]={0,2,6,8,12,14,18,20,24,26,30,32,36,38,42,44,
X                      48,50,54,56,60,62,66,68,72,74,78,80,84,86,90,92,96};
X    char lastpath[64];
X    char fs_insel[40];
X    char fs_inpath[40];
X    char *cutbuffer = NULL;
X    long cutlength = 0L;
X    int      ins  = FALSE;
X    int      inhex = TRUE;
X    int      partial = FALSE;
X    int      draw_flag = FALSE;
X    int      slid_flag = FALSE;
X    int      arro_flag = FALSE;
X#else
Xextern       char fsize_str[];
Xextern       char pos_str[];
Xextern       char s_str[];
Xextern       char r_str[];
Xextern       char dr_str[];
Xextern       int      ins;
Xextern       int      inhex;
Xextern       int      partial;
Xextern       int      draw_flag;
Xextern       int      slid_flag;
Xextern       int      arro_flag;
Xextern       char *cutbuffer;
Xextern       long cutlength;         /* markbuf end pointer
             */
Xextern       int  xoffset[];
Xextern       char     lastpath[];
Xextern       char fs_insel[];
Xextern       char fs_inpath[];
X#endif
SHAR_EOF
chmod 0600 GLOBALS.H || echo "restore of GLOBALS.H fails"
sed 's/^X//' << 'SHAR_EOF' > INIT.C &&
X
X#include <osbind.h>
X#include <gemdefs.h>
X#include <obdefs.h>
X
X#include "xxed.h" /* header file created by RCP     */
X#include "globals.h"  /* contains definition of menubar */
X
X
```

```
Xinit_resources()  /* loads resource array */
X{
X/*  if (!rsrc_load("xxed.rsc")) {
X        form_alert(1, "[0][Cannot find xxed.rsc file|Terminating ...][OK]");
X        exit(2);
X    }
X*/
X        mrsrc_load(XXEDRSC);  /* using .C RSC */
X
X}
X
Xinit_menu()  /* sets up menu bar  */
X{
X    rsrc_gaddr(0, MENUBAR, &menubar);
X    menubar[FSIZE].ob_spec = fsize_str;
X    menubar[POS].ob_spec = pos_str;
X    menu_bar(menubar, 1);
X}
X
Xinit_dialog()  /* sets up dialog boxes  */
X{
X    rsrc_gaddr(0, DIALOG2, &dialog2);
X    rsrc_gaddr(0, DIALOG1, &dialog1);
X    rsrc_gaddr(0, DIALOG3, &dialog3);
X    rsrc_gaddr(0, DIALOG4, &dialog4);
X    rsrc_gaddr(0, DIALOG5, &dialog5);
X    rsrc_gaddr(0, DIALOG6, &dialog6);
X
X    ((TEDINFO *)dialog1[SSTRING].ob_spec)->te_ptext = s_str;
X    ((TEDINFO *)dialog1[SSTRING].ob_spec)->te_txtlen = 24;
X    ((TEDINFO *)dialog1[RSTRING].ob_spec)->te_ptext = r_str;
X    ((TEDINFO *)dialog1[RSTRING].ob_spec)->te_txtlen = 24;
X    dialog4[DRSTR1].ob_spec = dr_str;
X    dialog1[SSTRBUT].ob_state = SELECTED;
X    dialog1[RSTRBUT].ob_state = SELECTED;
X
X}
X
Xinit_path()  /* initialize fs_inpath with current path */
X{
X    int drv;
X
X    drv = Dgetdrv();
X    Dgetpath(lastpath,0);
X    sprintf(fs_inpath,"%c:%s\\*.*",'A' + drv,lastpath);
X    strcpy(fs_insel,"untitled");
X}
X
X/*
X    shutdown - is the code that closes down the application.  This routine
X         is called when errors occurs and guarantees that all window's will
X         be closed properly before exiting.
X*/
```

```
Xshutdown(code)
X    int code;
X{
X    /*
X         Clean up memory.
X    */
X    cleanup();
X
X    /*
X         Clean up the io buffer because of bug in TOS ?
X    */
X
X    while(Cconis()) Crawcin();
X
X    /*
X         Shut down the application.
X    */
X    appl_exit();
X
X    /*
X         bye ...
X    */
X    exit(code);
X}
X
X
X/*
X    cleanup - releases the memory used by the application.
X*/
Xcleanup()
X{
X    windowptr thewin;
X
X    /*
X         Close down the windows.
X    */
X    for (thewin = firstwindow; thewin; thewin = thewin -> next)
X    {    dispose_buf(thewin);
X         dispose_window(thewin);
X    }
X
X    /*
X         free cutbuffer space
X    */
X
X    if(cutbuffer) free(cutbuffer);
X
X
X    /*
X         Free memory used by resource.
X
X    rsrc_free();  not needed now as using .C RSC
X
```

```
X    */
X}
X
X
X/*
X    open_vwork - Open a virtual workstation.
X
X    Note: a virtual workstation is associated with each window created.
X         This means that each window's graphic attributes are independent
X         of the other's.
X*/
Xint open_vwork(form)
X    register MFDB *form;
X{
X    register int x;
X    int   work_in[11];
X    int       handle, d;
X    int       work_out[57];
X
X    /*
X         Initialize workstation variables.
X    */
X    for(x=0; x<10; x++)
X         work_in[x] = 1;
X
X    work_in[10] = 2;
X
X    handle = graf_handle(&d, &d, &d, &d);
X    v_opnvwk(work_in, &handle, work_out);
X
X    form -> fd_addr    = Logbase();
X    form -> fd_w       = work_out[0] + 1;
X    form -> fd_h       = work_out[1] + 1;
X    form -> fd_wdwidth = form -> fd_w / 16;
X    form -> fd_stand   = 0;
X
X    switch(work_out[13]) {
X         case 16: form -> fd_nplanes = 4; break;
X         case 08: form -> fd_nplanes = 3; break;
X         case 04: form -> fd_nplanes = 2; break;
X         default: form -> fd_nplanes = 1; break;
X    }
X
X    return handle;
X}
SHAR_EOF
chmod 0600 INIT.C || echo "restore of INIT.C fails"
sed 's/^X//' << 'SHAR_EOF' > KEYS.C &&
X
X#include <gemdefs.h>
X#include <obdefs.h>
X#include <osbind.h>
X
```

```
X#include "globals.h"
X#include "xxed.h"
X
Xlong saveb;
X
X
Xdo_kbd(keycode,keymods)
X    int  keycode,keymods;
X{
X    windowptr thewin;
X    char          inbuf[8];
X    long     lnum;
X    int           skip,button;
X
X/* this fuction is probably the most tangled code in the whole program */
X
X
X    if(inhex)
X         if(keycode == 0x1011)
X              handle_file(QUIT);
X
X    if (firstwindow == NULL) return; /* no window open  */
X
X    thewin = thefrontwin;
X
X    if(thewin->markson)
X         handle_marks(CLEAR);
X
X    graf_mouse(M_OFF, 0L);
X    skip = TRUE;
X    if(thewin->icount!=0) /* in the middle of a hex num input */
X         do_kbd2(thewin,keycode,keymods); /* handle hex input */
X    else
X    {    saveb = getbyte(thewin,thewin->position);
X                   /* store the current undirtied byte  */
X                   /* then check for special keys       */
X    switch (keycode) {
X         case 0x5200:  ins ^= TRUE;
X                             if(ins)
X                        {    menubar[INSERT].ob_state = CHECKED;
X                                  menubar[REPLACE].ob_state = NORMAL;
X                             }
X                             else
X                             {    menubar[INSERT].ob_state = NORMAL;
X                                  menubar[REPLACE].ob_state = CHECKED;
X                             }
X                             wr_ins_rpl(ins);
X                             break;
X         case 0x4d00:  cur_right(thewin);
X                             break;
X         case 0x5000:  cur_down(thewin);
X                             break;
X         case 0x4800:  cur_up(thewin);
```

```
X                            break;
X         case 0x4b00:  cur_left(thewin);
X                            break;
X         case 0x537f:  delete_one(thewin,thewin->position,1);
X                            if(repos(thewin))
X                                 partial = FALSE;
X                            else
X                                 partial = TRUE;
X                            send_redraw(thewin);
X                            break;
X         case 0x0e08:  if(thewin->position-1 >= 0)
X                            {    delete_one(thewin,--thewin->position,1);
X                                 if(repos(thewin))
X                                      partial = FALSE;
X                                 else
X                                      partial = TRUE;
X                                 one_page(thewin,1);
X                            }
X                            break;
X         default: skip = FALSE;
X                       }
X    if (!(skip))  /* if it was not a special key   */
X         {
X         if(inhex)
X              do_kbd2(thewin,keycode,keymods); /* handle hex input */
X         else      /* handle ascii input  */
X              {    if( (thewin->position==thewin->flen-1) && !ins) /* in replace
mode on terminal dummy byte */
X                   {    button = form_alert(1, "[1][ The last byte in the file is |
a dummy byte and cannot be | changed. It is not saved | with the file. Use insert
mode. ][OK]");
X                        thewin->icount = 0;
X                        graf_mouse(M_ON, 0L);
X                        return;
X                   }
X                        if(keycode & 0xff)
X                             {    sprintf(inbuf,"%x",(keycode & 0xff));
X                             inbuf[2] = 0;
X                             lnum = (long)keycode & 0xff;
X                                  enter_it(thewin,inbuf,lnum);
X                             }
X              }
X         }
X    } /* end of else for if(thewin->icount!=0).....  */
X    graf_mouse(M_ON, 0L);
X
X}
X
Xdo_kbd2(thewin,keycode,keymods) /* handles hex input */
X    windowptr thewin;
X    int  keycode,keymods;
X{
X
```

```
X    if(thewin->icount==0)  /* not in the middle of entering a hex number */
X   {switch (keycode) {
X
X        case 0x3002:  handle_search(BACK);
X                      break;
X        case 0x1e01:  handle_search(FORWARD);
X                      break;
X        case 0x2106:  graf_mouse(M_ON, 0L);
X                      handle_search(FIND);
X                      graf_mouse(M_OFF, 0L);
X                      break;
X        case 0x1910:  handle_marks(CLEAR);
X                      break;
X        case 0x180f:  handle_marks(END);
X                      break;
X        case 0x1709:  handle_marks(START);
X                      break;
X        case 0x1205:  handle_edit(ERASE);
X                      break;
X        case 0x2f16:  handle_edit(PASTE);
X                      break;
X        case 0x2e03:  handle_edit(COPY);
X                      break;
X        case 0x2d18:  handle_edit(CUT);
X                      break;
X        case 0x1011:  handle_file(QUIT);
X                      break;
X        case 0x1f13:  handle_file(SAVE);
X                      break;
X        default     : thewin->input[thewin->icount++] = keycode & 0xFF;
X                      if(thewin->markson) clear_marks(thewin);
X                      ins_rpl(thewin);
X                     }
X    }
X    else {    thewin->input[thewin->icount++] = keycode & 0xFF;
X                      ins_rpl(thewin);
X            }
X
X}
X
Xcur_up(thewin)
X    windowptr thewin;
X{
X     if((thewin->position-=16)<0)
X         thewin->position+=16;
X         else check_scroll(thewin);
X}
X
X
Xcur_down(thewin)
X    windowptr thewin;
X{
X     if((thewin->position+=16)>thewin->flen-1)
```

```
X         thewin->position-=16;
X         else check_scroll(thewin);
X}
X
Xcur_right(thewin)
X    windowptr thewin;
X{
X         if(++thewin->position>thewin->flen-1)
X              thewin->position--;
X              else check_scroll(thewin);
X}
X
Xcur_left(thewin)
X    windowptr thewin;
X{
X    if(--thewin->position<0)
X         thewin->position++;
X         else check_scroll(thewin);
X}
X
Xcheck_scroll(thewin) /* cursor off screen?? scroll or reposition */
X    windowptr thewin;
X{
X    long winlines, bottomchar;
X    int up, down;
X
X    winlines = thewin->work.g_h/gl_hchar; /* no of lines can put in window  */
X    bottomchar = thewin->topchar + (winlines * 16) - 1;
X
X    up = 2; down = 3;
X
X    if (thewin->position > bottomchar )
X         if (thewin->position > bottomchar +16)
X              {    jump_pos(thewin);
X                   send_redraw(thewin);
X              }
X         else send_arrow(thewin,down);
X    if (thewin->position < thewin->topchar)
X         if (thewin->position < thewin->topchar - 16)
X              {    jump_pos(thewin);
X                   send_redraw(thewin);
X              }
X         else send_arrow(thewin,up);
X
X}
X
X
Xrepos(thewin)  /* reposition and redraw with cursor on screen */
X    windowptr thewin;
X{
X    long winlines, bottomchar;
X
X    winlines = thewin->work.g_h/gl_hchar; /* no of lines can put in window  */
```

```
X    bottomchar = thewin->topchar + (winlines * 16) - 1;
X
X    if ( (thewin->position > bottomchar )||
X              (thewin->position < thewin->topchar) )
X    {    jump_pos(thewin);
X         return(1);
X    }
X    else
X         return(0);
X}
X
Xjump_pos(thewin) /* reset display parameters so cursor on screen */
X    windowptr thewin;
X{
X    long totallines, winlines, topwlnmax, topwline,dummy;
X    int       vslide;
X
X
X    totallines = 1 + thewin->flen/16;
X    winlines = thewin->work.g_h/gl_hchar; /* no of lines can put in window  */
X
X    thewin->topchar = ((thewin->position - (8 * winlines)) & 0xfffffff0);
X    if (thewin->topchar < 0)
X         thewin->topchar = 0;
X
X    topwlnmax = ((totallines - winlines)>0) ? (totallines-winlines) : 0;
X    topwline = (thewin->topchar/16 < topwlnmax) ? thewin->topchar/16 : topwlnmax;
X    vslide = (1000*topwline)/topwlnmax;
X    thewin->vslidepos = vslide;
X}
X
Xins_rpl(thewin)  /* enters hex characes into the file */
X    windowptr thewin;
X{
X    int       x,y,handle,button;
X    char *inbuf;
X    long lnum;
X    unsigned  b;
X
X    x =  thewin->xcur;
X    y =  thewin->ycur + gl_hchar-1;
X    handle = thewin->graf.handle;
X    inbuf = thewin->input;
X
X         inbuf[thewin->icount]='\0';
X    if( (thewin->position==thewin->flen-1) && !ins) /* in replace mode on
terminal dummy byte */
X    {    button = form_alert(1, "[1][ The last byte in the file is | a dummy byte
and cannot be | changed. It is not saved | with the file. Use insert mode. ]
[OK]");
X         thewin->icount = 0;
X         return;
X    }
```

```
X         if (hex_chk(thewin,inbuf[thewin->icount-1])==1) /* is a hex char */
X              if (thewin->icount==1) /* first hex char coming   */
X                   {    b = saveb;    /* saveb is the current byte
     */
X                        b = b & 0xf;
X                        lnum = strtol(thewin->input,(char **)NULL,16);
X                        lnum = (lnum<<4) + b;
X                        /* write it back with top nibble changed
     */
X                        putbyte(thewin,thewin->position,lnum);
X                        v_gtext(handle, x, y, inbuf);
X                   }
X              else
X    /* two valid hex characters entered. Enter in buffer and redrw line */
X              {    lnum = strtol(thewin->input,(char **)NULL,16);
X                   enter_it(thewin,inbuf,lnum);
X              }
X
X}
X
Xenter_it(thewin,inbuf,lnum) /* insert or replace byte redrawing screen */
X    windowptr thewin;
X    char      *inbuf;
X    long      lnum;
X
X{   int x,y,handle;
X
X                   handle = thewin->graf.handle;
X                   x =  thewin->xcur;
X                   y =  thewin->ycur + gl_hchar-1;
X                   v_gtext(handle, x, y, inbuf);
X                   thewin->icount=0;
X              if(!ins)  /* in replace mode */
X              {
X                        putbyte(thewin,thewin->position,lnum);
X                        graf_mouse(M_OFF, 0L);
X                        one_line2(thewin,thewin->position);
     /* see onepage.c */
X                        cur_right(thewin);
X                        graf_mouse(M_ON, 0L);
X              }
X              else /* in insert mode  */
X              {    inbuf = "  ";
X                        *inbuf = (char)lnum;
X                             /* restore current byte as HEX input dirties it */
X                        putbyte(thewin,thewin->position,saveb);
X                        insert_it(thewin,1L,inbuf);  /* see edit.c
     */
X                        graf_mouse(M_OFF, 0L);
X                   if(thewin->position == thewin->flen-1)
X                             one_line2(thewin,thewin->position);
X                   else
X                   {
```

```
X                       one_line2(thewin,thewin->position);
     /* see onepage.c */
X                       partial = TRUE;
X                       send_redraw(thewin);
X                  }
X                       graf_mouse(M_ON, 0L);
X                       cur_right(thewin);
X             }
X}
X
X
Xhex_chk(thewin, ch) /* checks to see if a hex character was entered */
X    windowptr thewin;
X    char ch;
X{
X    int ret, button;
X    static int times;
X    switch (ch) {
X         case '0': case '1': case '2': case '3': case '4': case '5':
X         case '6': case '7': case '8': case '9': case 'A': case 'B':
SHAR_EOF
echo "End of part 1, continue with part 2"
echo "2" > s2_seq_.tmp
exit 0


From 4923317092821912823
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,78d78256168c7ffb,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-11-11 00:03:40 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!wuarchive!gem.mps.ohio-
state.edu!usc!apple!sun-barr!newstop!sun!sunfedcomm!grapevine!
koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i006: xxed -- Hex/ASCII file editor  part02/03
Keywords: shar, C
Message-ID: <34745@grapevine.uucp>
Date: 11 Nov 89 08:03:40 GMT
Sender: daemon@sunfedcomm
Reply-To: charltn@ccu.umanitoba.ca (Jim Charlton)
Lines: 1910
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: charltn@ccu.umanitoba.ca (Jim Charlton)
Posting-number: Volume 3, Issue 6
Archive-name: xxed/part02

#!/bin/sh
# this is part 2 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file KEYS.C continued
```

```
#
CurArch=2
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
sed 's/^X//' << 'SHAR_EOF' >> KEYS.C
X        case 'C': case 'D': case 'E': case 'F': case 'a': case 'b':
X        case 'c': case 'd': case 'e': case 'f': ret = 1 ; times = 0; break;
X        case  8 :     thewin->icount = 0;  /* backspace  after one hex char */
X                      graf_mouse(M_OFF, 0L);
X        /*   restore the byte and redraw line */
X                      putbyte(thewin,thewin->position,saveb);
X                      one_line2(thewin,thewin->position);
X                      graf_mouse(M_ON, 0L);
X                      times = 0;
X                      ret = 0;
X                      break;
X        default : Bconout(2,7);
X                      thewin->icount--; /*  wipe out nonhex char  */
X                      times++;
X                      if (times>2)
X    button = form_alert(1, "[1][ | | Enter a HEX digit or Backspace ][ OK ]");

X                      ret = 0;
X                }
X        return(ret);
X}
X
Xwr_ins_rpl(insflag) /* writes insert/replace/hex/ascii on menu line */
X    int  insflag;
X{   windowptr thewin;
X
X    cursor(60,0);
X    if(thewin = thefrontwin)
X    {    if(insflag && inhex)
X              {printf("%s\n","HEX   insert ");}
X         if(!(insflag) && inhex)
X              {printf("%s\n","HEX   replace");}
X         if(insflag && !(inhex))
X              {printf("%s\n","ASCII insert ");}
X         if(!(insflag) && !(inhex))
X              {printf("%s\n","ASCII replace");}
X    }
X    else
X         printf("%s\n","             ");
X}
X
```

```
Xcursor(x,y)   /* position the cursor*/
X        int x,y;
X  {     Bconout(2,27); Bconout(2,'Y');
X        Bconout(2,(32 + y)); Bconout(2,(32 + x));
X  }
SHAR_EOF
chmod 0600 KEYS.C || echo "restore of KEYS.C fails"
sed 's/^X//' << 'SHAR_EOF' > MAIN.C &&
X#include <gemdefs.h>
X#include <osbind.h>
X#include <obdefs.h>
X
X#define  extern
X#include "globals.h"
X#undef   extern
X
X
X/*
X    main - executes initialization code and the starts the program.
X*/
Xmain()
X{
X    int  dummy, button;
X
X    /*
X         Initiailize the ROMs.
X    */
X    gl_apid = appl_init();
X
X    if(Getrez() == 0)
X         {    button = form_alert(1, "[1][ Sorry....    | Need medium or high
resolution. ][OK]");
X         shutdown(2);
X         }
X
X    /*
X         Load resources.
X    */
X    init_resources();
X
X    /*
X         Read menu resource, draw menu bar, read dialog resources.
X    */
X    init_menu();
X    init_dialog();
X    init_path();
X    /*
X         Get the Physical work station handle.
X    */
X    phys_handle = graf_handle(&gl_wchar, &gl_hchar, &dummy, &dummy);
X
X
X    /*
```

```
X        Initialize the mouse.
X    */
X    graf_mouse(ARROW, NULL);
X
X    set_table();  /*  makes two byte ascii hex table   */
X
X
X    /*
X        Handle events for application.
X    */
X    TaskMaster();
X
X    /*
X        bye...  Note: This will never be executed.
X    */
X    shutdown(0);
X}
X
Xset_table()  /* makes a table of ASCII hex bytes for 0 to 255  */
X{   unsigned i;
X
X    table[0] = '\0';
X    for(i=0;i<256;i++)
X        sprintf(table+2*i,"%02x",i);
X}
SHAR_EOF
chmod 0600 MAIN.C || echo "restore of MAIN.C fails"
sed 's/^X//' << 'SHAR_EOF' > MAKEFILE &&
X#
X# Environment variable CC should be set to path to cc.ttp
X#
X
XOFILES= main.o init.o events.o wind.o menu.o files.o\
X        onepage.o bufman.o slider.o cursor.o keys.o button.o\
X        edit.o search.o    send.o printer.o xxedrsc.o
X
Xxxed.prg : $(OFILES)
X    $(CC) $(OFILES) -o xxed.prg
X
X$(OFILES) : globals.h xxed.h
SHAR_EOF
chmod 0600 MAKEFILE || echo "restore of MAKEFILE fails"
sed 's/^X//' << 'SHAR_EOF' > MENU.C &&
X
X#include <gemdefs.h>
X#include <obdefs.h>
X#include <osbind.h>
X
X#include "xxed.h"      /* contains definition of menubar */
X#include "globals.h"   /* header file created by RCP     */
X
Xextern char *chk_sstr_len();
Xextern char *chk_rstr_len();
```

```
X
X/*
X    do_menu - determines which menu was selected and calls the
X         appropriate routine to handle the item selected.
X*/
Xdo_menu(message)
X    int *message;
X{
X    int menuid, itemid;
X
X    menuid = message[3];
X    itemid = message[4];
X
X    switch(menuid) {
X         case DESK:
X                   handle_desk(itemid);
X                   break;
X
X         case FILE:
X                   handle_file(itemid);
X                   break;
X
X         case EDIT:    handle_edit(itemid);
X                       break;
X         case MARK:    handle_marks(itemid);
X                       break;
X         case SEARCH:  handle_search(itemid);
X                            break;
X         case WINDOWS: rot_wind();
X    }
X
X    menu_tnormal(menubar, menuid, 1);
X}
X
Xhandle_desk(itemid)
X    int itemid;
X{
X    switch(itemid) {
X         case ABOUT:   do_dialog(dialog2);
X                       dialog2[OK1].ob_state = NORMAL;
X                       break;
X    }
X}
X
X
X/*
X    handle_file - performs the appropriate action for the menu item selected.
X*/
Xhandle_file(itemid)
X    int itemid;
X{
X    int  button;
X    windowptr thewin;
```

```
X
X    switch(itemid) {
X         case NEW: new();
X                   break;
X         case OPEN:
X              thewin = new_window(UPARROW | DNARROW | VSLIDE | SIZER
X                                       | MOVER | FULLER | CLOSER | NAME);
X
X              if(thewin==NULL) break; /* abort if no more window handles */
X
X              if   (button = getfile(thewin))
X                   {    if   (read_file(thewin))
X                             open_window(thewin);
X                        else
X                             dispose_win_resources(thewin);
X                   }
X              else
X                   dispose_win_resources(thewin);
X                             break;
X         case SAVE:          if(thewin=thefrontwin)
X                             save_file(thewin);
X                             break;
X         case SAVEAS:   if(thewin=thefrontwin)
X                             write_file(thewin);
X                             break;
X         case CLOSE:         if (thewin=thefrontwin)
X                             {    dispose_buf(thewin);
X                             dispose_window(thewin);
X                             }
X                             break;
X         case DELETE:   delfile();
X                             break;
X
X         case SPACE:         drives();
X                             break;
X
X         case PRINT:         if(thewin = thefrontwin)
X                             print(thewin);
X                             break;
X
X         case QUIT:          shutdown(0);
X                             break;
X    }
X}
X
Xnew()  /* opens a new window with associated file  */
X{
X    int  button;
X    linkbufptr    bufptr;
X    windowptr thewin;
X
X              thewin = new_window(UPARROW | DNARROW | VSLIDE | SIZER
X                                       | MOVER | FULLER | CLOSER | NAME);
```

```
X             if(thewin==NULL)
X                  return; /* abort if no more window handles */
X            bufptr = addmember(thewin);
X             if (bufptr == NULL)
X                  {    button = form_alert(1, "[1][ Out of memory. | Cannot create
new workspace ][OK]");
X                       dispose_win_resources(thewin);
X                  }
X             else
X             {    open_window(thewin);
X                  if(getfile(thewin))
X                  {    bufptr->inuse=1;
X                       thewin->flen =1;
X                       ins = TRUE;
X                       menubar[INSERT].ob_state = CHECKED;
X                       menubar[REPLACE].ob_state = NORMAL;
X                       wr_ins_rpl(ins);
X                  }
X                  else
X                  {    dispose_buf(thewin);
X                       dispose_window(thewin);
X                  }
X             }
X}
X
X/*
X    handle_mark - performs the appropriate action for the menu item selected.
X*/
Xhandle_marks(itemid)
X    int itemid;
X{
X    char string[80];
X    windowptr thewin;
X
X    switch(itemid) {
X         case START    :   thewin = thefrontwin;
X                           if(thewin && thewin->flen)/* a window is open ?  */
X                           start_mark(thewin);
X                           break;
X
X
X         case END  :    thewin = thefrontwin;
X                           if(thewin && thewin->flen)/* a window is open ?  */
X                           end_mark(thewin);
X                           break;
X
X         case CLEAR    :    thewin = thefrontwin;
X                           if(thewin && thewin->flen)/* a window is open ?  */
X                           clear_marks(thewin);
X                           break;
X
X
X    }
```

```
X
X}
X
Xstart_mark(thewin)
Xwindowptr    thewin;
X{   thewin->startmark = thewin->position;
X    if( (thewin->startmark <= thewin->endmark)
X                   || thewin->markson )
X    {    send_redraw(thewin);
X         thewin->markson = TRUE;
X         menubar[CUT].ob_state = NORMAL;
X         menubar[COPY].ob_state = NORMAL;
X         menubar[ERASE].ob_state = NORMAL;
X    }
X}
X
Xend_mark(thewin)
Xwindowptr    thewin;
X{
X    thewin->endmark = thewin->position<thewin->flen-1 ?
X                                   thewin->position : thewin->flen-2;
X    if( (thewin->endmark >= thewin->startmark)
X                   || thewin->markson )
X    {    send_redraw(thewin);
X         thewin->markson = TRUE;
X         menubar[CUT].ob_state = NORMAL;
X         menubar[COPY].ob_state = NORMAL;
X         menubar[ERASE].ob_state = NORMAL;
X    }
X}
X
Xclear_marks(thewin)
Xwindowptr    thewin;
X    {    thewin->startmark = 1;
X         thewin->endmark = 0;
X         one_page(thewin,0);
X         thewin->markson = FALSE;
X         menubar[CUT].ob_state = DISABLED;
X         menubar[COPY].ob_state = DISABLED;
X         menubar[ERASE].ob_state = DISABLED;
X    }
X
X
Xhandle_edit(itemid)
X    int itemid;
X{
X    char string[80];
X    windowptr thewin;
X    linkbufptr    amem;
X    long pos;
X
X    switch(itemid) {
X         case COPY:    if( (thewin = thefrontwin) && thewin->markson)
```

```
X                           {    copy(thewin);
X                                handle_marks(CLEAR);
X                                menubar[CUT].ob_state = DISABLED;
X                                menubar[COPY].ob_state = DISABLED;
X                                menubar[ERASE].ob_state = DISABLED;
X                                menubar[PASTE].ob_state = NORMAL;
X                           }
X                           break;
X
X
X       case CUT  :    if( (thewin = thefrontwin) && thewin->markson)
X                           {    copy(thewin);
X                                cutit(thewin);
X                                send_vslid(thewin);
X                                thewin->changed = TRUE;
X                                handle_marks(CLEAR);
X                                menubar[CUT].ob_state = DISABLED;
X                                menubar[COPY].ob_state = DISABLED;
X                                menubar[ERASE].ob_state = DISABLED;
X                                menubar[PASTE].ob_state = NORMAL;
X                           }
X                           break;
X
X       case PASTE    :    if(thewin = thefrontwin)
X                                paste(thewin);
X                                thewin->changed = TRUE;
X                           break;
X       case ERASE    :    if( (thewin = thefrontwin) && thewin->markson)

X                           {    cutit(thewin);
X                                send_vslid(thewin);
X                                thewin->changed = TRUE;
X                                handle_marks(CLEAR);
X                           }
X                           break;
X       case INSERT : ins = TRUE;
X                           wr_ins_rpl(ins);
X                          menubar[INSERT].ob_state = CHECKED;
X                           menubar[REPLACE].ob_state = NORMAL;
X                           break;
X       case REPLACE :     ins = FALSE;
X                           wr_ins_rpl(ins);
X                          menubar[INSERT].ob_state = NORMAL;
X                           menubar[REPLACE].ob_state = CHECKED;
X                           break;
X
X
X   }
X
X}
X
Xhandle_search(itemid)
X   int itemid;
```

```
X{
X    char *sstr;
X    windowptr thewin;
X
X    if(thewin=thefrontwin)
X    {    switch(itemid) {
X              case FIND :         find0(thewin);
X                                       break;
X              case FORWARD  :     if(sstr = chk_sstr_len(thewin))
X                                       forward1(thewin,sstr);
X                                       check_scroll(thewin);
X                                       break;
X              case BACK :         if(sstr = chk_sstr_len(thewin))
X                                       back1(thewin,sstr);
X                                       break;
X                             }
X    }
X}
SHAR_EOF
chmod 0600 MENU.C || echo "restore of MENU.C fails"
sed 's/^X//' << 'SHAR_EOF' > ONEPAGE.C &&
X#include "stdio.h"
X#include "gemdefs.h"
X#include "obdefs.h"
X#include "osbind.h"
X
X#include "globals.h"
X
X
X    unsigned getbyte();
X
X    char str2[80],str3[80];
X    char twoblanks[5]="  ";
X
X/*      Display one window full of data.       */
X
Xone_page(thewin, blank) /* draw one window full of data */
X    windowptr thewin;
X    int           blank;
X{   int i, nc;
X
X    wr_ins_rpl(ins); /* input status written upper rt.  see keys.c */
X
X    graf_mouse(M_OFF, 0L);
X    nc = (thewin->work.g_h/gl_hchar)*16;  /* nc is number of char */
X    if(partial)
X    {    nc = nc - (thewin->position - thewin->topchar ) + 16;
X         for(i=0;i<nc;i+=16)
X              one_line0(thewin,thewin->position+i);
X              partial = FALSE;
X    }
X    else
X    {
```

```
X         wind_blank(thewin);
X         for(i=0;i<nc;i+=16)
X              one_line0(thewin,thewin->topchar+i);
X    }
X         graf_mouse(M_ON, 0L);
X
X}
X
X
Xunsigned getbyte(thewin,pos) /* retreive any byte in file in ram */
X    windowptr thewin;
X    long pos;
X{
X    linkbufptr amem;
X    char *addr;
X
X    if (pos<thewin->flen)
X    {    amem = thewin->headptr;
X         while (amem->inuse<=pos)
X              {    pos = pos - amem->inuse;
X                   amem = amem->next;
X              }
X         addr = (char *)(amem->block+pos);
X         return(*addr);
X    }
X    else return((unsigned)0x00);
X}
X
Xputbyte(thewin,pos,lnum) /* put byte lnum into file at pos */
X    windowptr thewin;
X    long pos, lnum;
X{
X    linkbufptr amem;
X    char *addr;
X
X    if (pos<thewin->flen)
X    {    amem = thewin->headptr;
X         while (amem->inuse<=pos)
X              {    pos = pos - amem->inuse;
X                   amem = amem->next;
X              }
X         addr = (char *)(amem->block+pos);
X         *addr = (char)lnum;
X         thewin->changed = TRUE;
X    }
X}
X
X
Xone_line0(thewin,pos)
X    windowptr thewin;
X    long          pos;
X{
X    long l;
```

```
X
X    l = pos&~0xf;
X    if( (l > thewin->endmark) || ((l+16) < thewin->startmark) )
X         one_line1(thewin,pos);
X    else
X         one_line2(thewin,pos);
X
X}
X
Xone_line2(thewin,pos) /* handles mixed marked and unmarked */
X    windowptr thewin;
X    long          pos;
X
X{   int i, lineno;
X    int           tx, ty;
X    int           grafhandle;
X    long  l;
X    unsigned num;
X    register int j,k;
X    char *ptr3;
X
X    tx = thewin -> work.g_x + 7;
X    ty = thewin -> work.g_y + gl_hchar;
X    grafhandle = thewin -> graf.handle;
X
X
X    lineno = (pos - thewin->topchar)>>4;
X    ty += (gl_hchar*lineno);
X              l = pos&~0xf;
X              ptr3=str3;
X              str2[2] = '\0';
X         for (j=0; j < 16 ; j +=2 )
X          {  for (k=0; k < 2  ; k++   )
X              {    if(l+j+k < thewin->flen-1)
X                   {    num = getbyte(thewin,l + j + k);
X                        if(num)
X                             *(ptr3++) = num;
X                        else
X                             *(ptr3++) = (unsigned)32;
X                        bcopy(table+num*2,str2,2);
X                        if( (l+j+k>=thewin->startmark) && (l+j+k<=thewin-
>endmark) )
X                             vswr_mode(thewin->graf.handle,4);
X                        else
X                             vswr_mode(thewin->graf.handle,1);
X                   }
X                   else
X                   {    bcopy(twoblanks,str2,3);
X                        *(ptr3++) = (unsigned)32;
X                   }
X                        v_gtext(grafhandle, tx, ty, str2);
X                   tx += 2*gl_wchar;
X              }
```

```
X                       vswr_mode(thewin->graf.handle,1);
X                       bcopy(twoblanks,str2,3);
X                       v_gtext(grafhandle, tx, ty, str2);
X                       tx += 2*gl_wchar;
X         }
X                   *(ptr3++)=0x0;
X                   vswr_mode(thewin->graf.handle,1);
X                   v_gtext(thewin->graf.handle, tx, ty, str3);
X}
X
Xone_line1(thewin,pos) /* handles unmarked text    */
X    windowptr thewin;
X    long          pos;
X
X{   int i, lineno;
X    int           tx, ty;
X    int           grafhandle;
X    long  l;
X    unsigned num;
X    register int j,k;
X    char *ptr2,*ptr3;
X
X    tx = thewin -> work.g_x + 7;
X    ty = thewin -> work.g_y + gl_hchar;
X
X    lineno = (pos - thewin->topchar)>>4;
X    ty += (gl_hchar*lineno);
X              l = pos&~0xf;
X              ptr2 = str2;
X              ptr3 = str3;
X
X                for (j=0; j < 16 ; j +=2 )
X           {  for (k=0; k < 2  ; k++   )
X             {     num = getbyte(thewin,l + j + k);
X              if(l+j+k < thewin->flen-1)
X                   {
X                        if(num)
X                             *(ptr3++) = num;
X                        else
X                             *(ptr3++) = (unsigned)32;
X                        bcopy(table+num*2,ptr2,2);
X                        ptr2+=2;
X                   }
X                   else
X                   {    bcopy(twoblanks,ptr2,2);
X                        ptr2+=2;
X                        *(ptr3++) = 0x20;
X                   }
X              }
X                   *(ptr2++)=0x20;
X                   *(ptr2++)=0x20;
X            }
X                   *(ptr2++)=0x20;
```

```
X                 *(ptr2++)=0x20;
X                 *(ptr2++)=0x0;
X                 v_gtext(thewin->graf.handle, tx, ty, str2);
X                 tx += 48*gl_wchar;
X                 *(ptr3++)=0x0;
X                 v_gtext(thewin->graf.handle, tx, ty,str3);
X}
X
SHAR_EOF
chmod 0600 ONEPAGE.C || echo "restore of ONEPAGE.C fails"
sed 's/^X//' << 'SHAR_EOF' > PRINTER.C &&
X#include <gemdefs.h>
X#include <obdefs.h>
X#include <osbind.h>
X#include <stdio.h>
X#include "xxed.h"
X
X#include "globals.h"
X
X    extern unsigned getbyte();
X    extern char str2[],str3[];
X    extern char twoblanks[];
X
X
Xprint(thewin) /* send file or block to printer */
Xwindowptr thewin;
X{
X    int button;
X    char str[200];
X    long l;
X
X    if(Bcostat(0)==0)
X    {    button = form_alert(1, "[1][ The printer is not connected | or not turned
on! ][ OK ]");
X         return;
X    }
X
X    while(Cconis()) Crawcin(); /* make sure io buffer clear */
X    if(thewin->markson)
X    {    sprintf(str, "[1][ Print block from | %s ? | (Any key aborts    |  while
printing) ][ OK | CANCEL ]",thewin->title);
X         button = form_alert(1,str);
X         if (button == 1)
X              for (l=thewin->startmark;l<thewin->endmark+16;l+=16)
X             {   print_line(thewin,l);
X                  if (Cconis())
X                  {    Crawcin();
X                       break;  /* any key to abort */
X                  }
X              }
X    }
X    else
X    {    sprintf(str, "[1][ Print file | %s ? | (Any key aborts    |  while
```

```
printing) ][ OK | CANCEL ]",thewin->title);
X         button = form_alert(1,str);
X         if (button == 1)
X              for (l=0;l<thewin->flen+17;l+=16)
X             {   print_line(thewin,l);
X                   if (Cconis())
X                   {    Crawcin();
X                        break;  /* any key to abort */
X                   }
X              }
X    }
X
X}
X
X
X
Xprint_line(thewin,pos) /* send a line to the printer */
X    windowptr thewin;
X    long      pos;
X{   int i;
X    long  l;
X    unsigned num;
X    register int j,k;
X    char *ptr2,*ptr3;
X
X              l = pos&~0xf;
X              ptr2 = str2;
X              ptr3 = str3;
X
X                for (j=0; j < 16 ; j +=2 )
X           {  for (k=0; k < 2  ; k++   )
X             {     num = getbyte(thewin,l + j + k);
X              if(l+j+k < thewin->flen-1)
X                   {
X                        if(num>31)
X                             *(ptr3++) = num;
X                        else
X                             *(ptr3++) = (unsigned)0x2E;
X                        bcopy(table+num*2,ptr2,2);
X                        ptr2+=2;
X                   }
X                   else
X                   {    bcopy(twoblanks,ptr2,2);
X                        ptr2+=2;
X                        *(ptr3++) = 0x20;
X                   }
X              }
X                   *(ptr2++)=0x20;
X                   *(ptr2++)=0x20;
X            }
X                   *(ptr2++)=0x20;
X                   *(ptr2++)=0x20;
X                   *(ptr2++)=0x0;
```

```
X                  pr_print(str2);
X                  *(ptr3++)=0x0D;
X                  *(ptr3++)=0x0A;
X                  *(ptr3++)=0x0;
X                  pr_print(str3);
X}
X
X
Xpr_print(str) /* send a string to the printer */
X    char *str;
X{   int i;
X
X    {    for (i=0;i<(strlen(str)+1);i++)
X              Bconout(0,str[i]);
X    }
X}
SHAR_EOF
chmod 0600 PRINTER.C || echo "restore of PRINTER.C fails"
sed 's/^X//' << 'SHAR_EOF' > SEARCH.C &&
X#include <stdio.h>
X#include <gemdefs.h>
X#include <obdefs.h>
X#include <osbind.h>
X
X#include "globals.h"
X#include "xxed.h"
X
Xlong forward2();
Xlong back2();
Xchar *chk_sstr_len();
Xchar *chk_rstr_len();
X
Xlong rl,sl;  /* length of replace and search strings  */
X
X
Xfind(thewin) /* handles search and replace dialog box  */
X    windowptr thewin;
X{
X    int  button;
X
X
X    button = do_dialog(dialog1);
X    dialog1[SFORWARD].ob_state = NORMAL;
X    dialog1[SBACK].ob_state = NORMAL;
X    dialog1[CANCEL1].ob_state = NORMAL;
X    dialog1[RONCE].ob_state = NORMAL;
X    dialog1[RALL].ob_state = NORMAL;
X    dialog1[RVER].ob_state = NORMAL;
X
X
X    return(button);
X}
X
```

```
Xchar *chk_sstr_len(thewin) /* check search string length */
X    windowptr thewin;
X{
X    char *sstr;
X    int  button;
X
X         if( (dialog1[SCUTBUF].ob_state == SELECTED) )
X         {    sstr = cutbuffer;
X              sl = cutlength;
X         }
X         else
X         {    sstr = s_str;
X              sl = strlen(sstr);
X         }
X              if ( (sstr == NULL) || (sl == 0) )
X    {    button = form_alert(1, "[1][ Nul search string... | Aborting.   ]
[ OK ]");
X         return(0);
X    }
X              else
X         return(sstr);
X}
X
Xchar *chk_rstr_len(thewin)
X    windowptr thewin;
X{
X    char *rstr;
X    int  button;
X
X         if( (dialog1[RCUTBUF].ob_state == SELECTED) )
X         {    rstr = cutbuffer;
X              rl = cutlength;
X         }
X         else
X         {    rstr = r_str;
X              rl = strlen(rstr);
X         }
X              if ( (rstr == NULL) || (rl == 0) )
X    {    button = form_alert(1, "[1][ Nul replace string... | Aborting.   ]
[ OK ]");
X         return(0);
X    }
X              else
X         return(rstr);
X}
X
Xfind0(thewin) /* handles results from srch/rpl dialog */
X    windowptr thewin;
X{
X    char *sstr, *rstr;
X    int  button;
X    long pos;
X
```

```
X        button = find(thewin);
X        if (button == CANCEL1)
X             return;
X        if (!(sstr = chk_sstr_len(thewin)))
X             return;
X
X        switch (button)
X             {
X             case SFORWARD :    forward1(thewin,sstr);
X                                check_scroll(thewin);
X                                break;
X             case SBACK :  back1(thewin,sstr);
X                                break;
X             case RONCE :  if (!(rstr = chk_rstr_len(thewin)))
X                                     break;
X                                ronce(thewin,sstr,rstr);
X                                repos(thewin);
X                                one_page(thewin,1);
X                                redraw_vslider(thewin->handle);
X                                break;
X             case RALL :        if (!(rstr = chk_rstr_len(thewin)))
X                                     break;
X                                rall(thewin,sstr,rstr);
X                                break;
X             case RVER :        if (!(rstr = chk_rstr_len(thewin)))
X                                     return;
X                                rver(thewin,sstr,rstr);
X                                break;
X             default :          ;
X             }
X}
X
Xforward1(thewin,sstr) /* searches forward */
X    windowptr thewin;
X    char *sstr;
X{
X    long pos;
X    int  button;
X
X    graf_mouse(2,0L);
X
X    if(thewin->position < thewin->flen-2) thewin->position += 1;
X    if( (pos=forward2(thewin,sstr)) > 0 )
X    {    thewin->position = pos;
X         graf_mouse(0,0L);
X         return(-1);
X    }
X    else
X    {    button = form_alert(1, "[1][ | | No (further) occurrences of | the string
can be found. |  ][ OK ]");
X         thewin->position -= 1;
X         graf_mouse(0,0L);
X         return(0);
```

```
X    }
X}
X
Xrp_forw(thewin,sstr) /* replace going forward */
X    windowptr thewin;
X    char *sstr;
X{
X    long pos;
X    int  button;
X
X    if( (pos=forward2(thewin,sstr)) > -1)
X    {    thewin->position = pos;
X         return(-1);
X    }
X    else
X    {    one_page(thewin,1);
X         redraw_vslider(thewin->handle);
X         button = form_alert(1, "[1][ | | No (further) occurrences of | the string
can be found. |  ][ OK ]");
X         return(0);
X    }
X}
X
X
Xlong forward2(thewin,sstr)
X    windowptr thewin;
X    char *sstr;
X{
X    unsigned  num;
X    int  j, strl;
X    long i;
X
X         strl = strlen(sstr);
X         for (i=thewin->position;i<(thewin->flen-strl);i++)
X         {    j=1;
X              num = getbyte(thewin,i);
X              if(num == sstr[0])
X              {    while (j<strl)
X                   {    if( (num = getbyte(thewin,i+j)) == sstr[j])
X                             j+=1;
X                        else
X                             break;
X                   }
X                   if (j == strl)
X                        return(i);
X              }
X         }
X         return(-1);
X}
X
Xback1(thewin,sstr) /* search backwards */
X    windowptr thewin;
X    char *sstr;
```

```
X{
X    long pos;
X    int  button;
X
X    graf_mouse(2,0L);
X
X         thewin->position -= 1;
X    if( (pos=back2(thewin,sstr)) > -1 )
X    {    thewin->position = pos;
X         check_scroll(thewin);
X    }
X    else
X    {    button = form_alert(1, "[1][ | |   String not found.   ][ OK ]");
X         thewin->position += 1;
X    }
X    graf_mouse(0,0L);
X}
X
Xlong back2(thewin,sstr)
X    windowptr thewin;
X    char *sstr;
X{
X    unsigned num;
X    int  j, strl;
X    long i;
X
X
X         strl = strlen(sstr);
X         for (i=thewin->position;i > -1;i--)
X         {    j=1;
X              num = getbyte(thewin,i);
X              if(num == sstr[0])
X              {    while (j<strl)
X                   {    if( (num = getbyte(thewin,i+j)) == sstr[j])
X                             j+=1;
X                        else
X                             break;
X                   }
X                   if (j == strl)
X                        return(i);
X              }
X         }
X         return(-1);
X}
X
Xronce(thewin,sstr,rstr) /* replace once */
X    windowptr thewin;
X    char *sstr, *rstr;
X{
X    long rl;
X    int sl, button, c;
X
X
```

```
X    if((c = strcmp(sstr,rstr))==0)
X    {    button = form_alert(1, "[1][ | Search and Replace strings | are
identical.  Aborting...  |  ][ OK ]");
X         return(0);
X    }
X    graf_mouse(2,0L);
X    if(rp_forw(thewin,sstr))
X    {    rl = strlen(rstr);
X         sl = strlen(sstr);
X         thewin->startmark = thewin->position;
X         thewin->position += sl;
X         thewin->endmark = thewin->position -1;
X         thewin->markson = TRUE;
X         cutit(thewin);
X         thewin->changed = TRUE;
X/* following code just clears the marks  */
X              {    thewin->startmark = 1;
X                   thewin->endmark = 0;
X                   thewin->markson = FALSE;
X              }
X
X    /* insert_it() inserts rl bytes from string pointed to by rstr
X    into the file before the position of the cursor... see edit.c
X    */
X         insert_it(thewin,rl,rstr);
X         graf_mouse(0,0L);
X         return(-1);
X    }
X    else
X         graf_mouse(0,0L);
X         return(0);
X}
X
Xrall(thewin,sstr,rstr) /* replace all */
X    windowptr thewin;
X    char *sstr, *rstr;
X{
X    int c, button;
X
X    if((c = strcmp(sstr,rstr))==0)
X    {    button = form_alert(1, "[1][ | Search and Replace strings | are
identical.  Aborting...  |  ][ OK ]");
X         return(0);
X    }
X    one_page(thewin,1);
X    while(ronce(thewin,sstr,rstr))
X         thewin->position += rl;
X    thewin->position -= rl;
X    jump_pos(thewin);
X    send_redraw(thewin);
X}
Xrver(thewin,sstr,rstr) /* replace with verify */
X    windowptr thewin;
```

```
X    char *sstr, *rstr;
X{
X    int button, result, c;
X
X    if((c = strcmp(sstr,rstr))==0)
X    {    button = form_alert(1, "[1][ | Search and Replace strings | are
identical.  Aborting...  |  ][ OK ]");
X         return(0);
X    }
X    graf_mouse(2,0L);
X    result = 1;
X    while(result)
X    {    if(rp_forw(thewin,sstr))
X         {    repos(thewin);
X              one_page(thewin,1);
X              redraw_vslider(thewin->handle);
X                   if(thewin = thefrontwin)
X                   {    graf_mouse(M_OFF, 0L);
X                        putcur(thewin);    /* cursor on  */
X                        graf_mouse(M_ON, 0L);
X                   }
X              graf_mouse(0,0L);
X              button = do_dialxy(dialog3,100,100);
X              graf_mouse(2,0L);
X              dialog3[RPBUT].ob_state = NORMAL;
X              dialog3[SKBUT].ob_state = NORMAL;
X              dialog3[ABBUT].ob_state = NORMAL;
X                   if(thewin = thefrontwin)
X                   {    graf_mouse(M_OFF, 0L);
X                        putcur(thewin);    /* cursor off */
X                        graf_mouse(M_ON, 0L);
X                   }
X         if(button == RPBUT)
X              {    thewin->startmark = thewin->position;
X                   thewin->position += sl;
X                   thewin->endmark = thewin->position -1;
X                   thewin->markson = TRUE;
X                   cutit(thewin);
X                   thewin->changed = TRUE;
X         /* following code just clears the marks  */
X                        {    thewin->startmark = 1;
X                             thewin->endmark = 0;
X                             thewin->markson = FALSE;
X                        }
X
X              /* insert_it() inserts rl bytes from string pointed to by rstr
X              into the file before the position of the cursor... see edit.c
X              */
X                   insert_it(thewin,rl,rstr);
X                   thewin->position += rl;
X                   result = 1;
X              }
X              if(button == SKBUT)
```

```
X                  if(thewin->position < thewin->flen-2)
X                           thewin->position += 1;
X             if(button == ABBUT)
X                  break;
X         } /* end of if(rp_forw())  */
X         else
X         {    thewin->position -= rl;
X              result = 0;
X         }
X    }  send_redraw(thewin);
X    graf_mouse(0,0L);
X}
X
X
Xdo_dialog(dialog)   /* draw dialog boxes   */
XOBJECT *dialog;
X{
X    int  cx, cy, cw, ch, button;
X
X    form_center(dialog, &cx, &cy, &cw, &ch);
X    form_dial(FMD_START, 0, 0, 0, 0, cx, cy, cw, ch);
X    objc_draw(dialog, 0, 10, cx, cy, cw, ch);
X    button = form_do(dialog, 0);
X    form_dial(FMD_FINISH, 0, 0, 0, 0, cx, cy, cw, ch);
X    return(button);
X}
Xdo_dialxy(dialog,x,y) /* draw dialog at specific position with form_do */
XOBJECT *dialog;
Xint x,y;
X{
X    int  cx, cy, cw, ch, button;
X
X    dialog->ob_x = x;
X    dialog->ob_y = y;
X    cx = x;
X    cy = y;
X    cw = dialog->ob_width;
X    ch = dialog->ob_height;
X
X    form_dial(FMD_START, 0, 0, 0, 0, cx, cy, cw, ch);
X    objc_draw(dialog, 0, 10, cx, cy, cw, ch);
X    button = form_do(dialog, 0);
X    form_dial(FMD_FINISH, 0, 0, 0, 0, cx, cy, cw, ch);
X    return(button);
X}
X
Xdrw_dialxy(dialog,x,y) /* draw object at a position.. no form_do */
XOBJECT *dialog;
Xint x,y;
X{
X    int  cx, cy, cw, ch;
X
X    dialog->ob_x = x;
```

```
X    dialog->ob_y = y;
X    cx = x;
X    cy = y;
X    cw = dialog->ob_width;
X    ch = dialog->ob_height;
X    form_dial(FMD_START, 0, 0, 0, 0, cx, cy, cw, ch);
X    objc_draw(dialog, 0, 10, cx, cy, cw, ch);
X}
SHAR_EOF
chmod 0600 SEARCH.C || echo "restore of SEARCH.C fails"
sed 's/^X//' << 'SHAR_EOF' > SEND.C &&
X
X#include "stdio.h"
X#include "gemdefs.h"
X#include "obdefs.h"
X#include "osbind.h"
X#include "globals.h"
X
Xsend_vslid(thewin) /* send application slider message */
X    windowptr thewin;
X{
X         int msg[8];
X                    /* the message buffer */
X
X    if(slid_flag)
X        return;
X/* return if there is already a vslid message pending  */
X
X         msg[0] = WM_VSLID;        /* message type is vslid */
X         msg[1] = gl_apid;          /* application id*/
X         msg[2] = 0;                /* message is standard 16 bytes */
X         msg[3] = thewin->handle;        /* handle of window to refresh */
X         msg[4] = thewin->vslidepos;       /* position of vslider */
X
X         appl_write( gl_apid, 16, msg );
X        slid_flag = TRUE;
X
X}
X
X
Xsend_redraw(thewin) /* send application a redraw message */
X    windowptr thewin;
X{
X         int msg[8];
X    if(draw_flag)
X        return;
X/* return if there is already a draw message pending   */
X
X                    /* the message buffer */
X         msg[0] = WM_REDRAW;        /* message type is redw */
X         msg[1] = gl_apid;          /* application id*/
X         msg[2] = 0;                /* message is standard 16 bytes */
X         msg[3] = thewin->handle;        /* handle of window to refresh */
```

```
X         msg[4] = thewin->work.g_x;            /* position and size of redraw */
X         msg[5] = thewin->work.g_y;
X         msg[6] = thewin->work.g_w;
X         msg[7] = thewin->work.g_h;
X
X         appl_write( gl_apid, 16, msg );
X        draw_flag = TRUE;
X
X/*
X This will send a redraw to your own application. The above routine is taken
Xfrom COMPUTES Technical Reference Guide for the Atari ST - Volume 2 AES. Have
Xfun.
X*/
X
X}
X
Ximmed_redraw(thewin) /* bypass event multi and do a screen redraw */
X    windowptr thewin;
X{
X         int msg[8];
X
X                    /* the message buffer */
X         msg[0] = WM_REDRAW;        /* message type is redw */
X         msg[1] = gl_apid;          /* application id*/
X         msg[2] = 0;                /* message is standard 16 bytes */
X         msg[3] = thewin->handle;        /* handle of window to refresh */
X         msg[4] = thewin->work.g_x;      /* position and size of redraw */
X         msg[5] = thewin->work.g_y;
X         msg[6] = thewin->work.g_w;
X         msg[7] = thewin->work.g_h;
X    do_window(msg);
X}
X
Xsend_arrow(thewin,direction) /* send application an arrow message */
X    windowptr thewin;
X    int           direction;
X{
X         int msg[8];
X                    /* the message buffer */
X
X    if(arro_flag)
X         return;
X/* return if there is already a arrow message pending  */
X
X         msg[0] = WM_ARROWED;        /* message type is */
X         msg[1] = gl_apid;          /* application id*/
X         msg[2] = 0;                /* message is standard 16 bytes */
X         msg[3] = thewin->handle;        /* handle of window to refresh */
X         msg[4] = direction;           /* up = 2, down = 3 */
X         msg[5] = 0;
X         msg[6] = 0;
X         msg[7] = 0;
X
```

```
X         appl_write( gl_apid, 16, msg );
X        arro_flag = TRUE;
X}
SHAR_EOF
chmod 0600 SEND.C || echo "restore of SEND.C fails"
sed 's/^X//' << 'SHAR_EOF' > SLIDER.C &&
X
X#include "stdio.h"
X#include "gemdefs.h"
X#include "obdefs.h"
X#include "osbind.h"
X#include "globals.h"
X
X
X/* takes care of displaying the correct text in the window after
X      the slider has been moved or the window resized  */
Xvhandler(message)
X    int *message;
X{   int  vslideold, vslide;
X    long totallines, topwlnmax, topwline, winlines, topchar;
X    int      handle, ret;
X    windowptr thewin;
X
X    handle = message[3];
X    thewin = findwindowptr(handle);
X
X
X    totallines = 1 + thewin->flen/16;
X
X    winlines = thewin->work.g_h/gl_hchar; /* no of lines can put in window  */
X
X    vslideold = thewin->vslidepos;  /* save the old slider position  */
X
X /* calculate the line number (from the top of the file) for the top
X    line displayed on the screen when the last full page of text
X    is being displayed in the window   */
X
X    topwlnmax = ((totallines - winlines)>0) ? (totallines-winlines) : 0;
X
X  /* topwline is the top line of the currently displayed page */
X
X    topwline = (thewin->topchar/16 < topwlnmax) ? thewin->topchar/16 : topwlnmax;
X
X    if(thewin->topchar > topwline*16)
X         send_redraw(thewin);
X    thewin->topchar = topwline*16;
X    if(topwlnmax)
X         thewin->vslidepos = (1000*topwline)/topwlnmax;
X    else
X         thewin->vslidepos = 1;
X
X  switch (message[0]) {
X    case WM_ARROWED    :
```

```
X            switch (message[4]) {
X                case 0 :    /* uppage */
X                  topwline = (topwline-winlines+1>0) ? topwline-winlines+1 : 0;
X                       thewin->topchar = topwline*16;
X                   if(topwline == 0) thewin->vslidepos = 1;
X                     else    thewin->vslidepos = (1000*topwline)/topwlnmax;
X                       send_redraw(thewin);
X                  break;
X                case 1 :     /* dnpage  */
X                  topwline += winlines - 1;
X                       if(topwline>topwlnmax)topwline = topwlnmax;
X                       thewin->topchar = topwline*16;
X                   if(topwline == 0) thewin->vslidepos = 1;
X                     else    thewin->vslidepos = (1000*topwline)/topwlnmax;
X                       send_redraw(thewin);
X                      break;
X                case 2 :     /* upline  */
X                       if(--topwline<0) topwline++;
X                       else
X                       {    thewin->topchar = topwline*16;
X                        if(topwline == 0) thewin->vslidepos = 1;
X                          else    thewin->vslidepos = (1000*topwline)/topwlnmax;
X                                 scroll(thewin,0,1);
X                                 redraw_vslider(thewin->handle);
X                       }
X                     break;
X                case 3 :     /* dnline  */
X                       if(++topwline>topwlnmax) topwline--;
X                       else
X                       {    thewin->topchar = topwline*16;
X                         if(topwline == 0) thewin->vslidepos = 1;
X                         else    thewin->vslidepos = (1000*topwline)/topwlnmax;
X                                 scroll(thewin,1,0);
X                                 redraw_vslider(thewin->handle);
X                       }
X                       break;         }  /* end of switch message[4]  */
X
X                       break; /* from WM_ARROWED */
X
X    case WM_VSLID :
X                /* the sliderbox was "pulled" up or down  */
X       vslide = message[4];
X           /* if the slider box was moved by less than 5 units do nothing */
X         if (abs(vslideold-vslide)<5)
X                  {    vslide = vslideold;
X                  }
X              else  {  topwline = vslide*topwlnmax/1000;
X                       thewin->topchar = topwline*16;
X                       thewin->vslidepos = vslide;
X                       send_redraw(thewin);
X                    }
X
X                  break; /* from WM_VSLID   */
```

```
X                       }    /* end of switch message[0]  */
X}
X
Xredraw_vslider(wihandle)    /* redraws vslider in new postion  */
X    int  wihandle;
X{
X    windowptr thewin;
X    int vslide_size, ret;
X    long totallines, seenlines;
X
X    thewin = findwindowptr(wihandle);
X
X    totallines = 1 + thewin->flen/16;
X
X    seenlines = min(totallines,(thewin->work.g_h/gl_hchar));
X    vslide_size = min(1000,1000*seenlines/totallines);
X
X  /* redraw the slider  to its new position and with new size */

X    wind_update(BEG_UPDATE);
X    wind_set(thewin->handle,WF_VSLSIZE,vslide_size,0,0,0);
X    wind_set(wihandle,WF_VSLIDE,thewin->vslidepos,&ret,&ret,&ret);
X    wind_update(END_UPDATE);
X
X}
X
Xsize_vslider(wihandle)  /* set slider size  and redraws it */
X    int wihandle;
X{   int vslide_size;
X    long totallines, seenlines;
X    windowptr thewin;
X
X    thewin = findwindowptr(wihandle);
X
X    totallines = 1 + thewin->flen/16;
X
X    seenlines = min(totallines,(thewin->work.g_h/gl_hchar));
X    vslide_size = min(1000,1000*seenlines/totallines);
X    wind_set(thewin->handle,WF_VSLSIZE,vslide_size,0,0,0);
X}
X
X   /* the scroll routine which moves text up if the up flag is set
X      and down if the dnflag is set.  The other flag must be zero.
X      Finally blanks the space for the new line at top or bottom
X      and sets the clipping rectangle for just that line in preparation
X      for the text in the window to be redrawn.  Be sure to reset
X      the clipping to the full window work area after redrawing the
X      text        */
X
Xscroll(thewin,upflag,dnflag)
X    windowptr thewin;
X   int upflag, dnflag;
X
```

```
X{   int pxy[8], pxyc[4], i, lines_onscrn;
X    int xwork, ywork, hwork, wwork;
X    long totallines,seenlines;
X
X        /* upflag=1 dnflag=0 for scroll up one line  */
X        /* upflag=0 dnflag=1 for scroll down one line  */
X
X    xwork = thewin->work.g_x;
X    ywork = thewin->work.g_y;
X    hwork = thewin->work.g_h;
X    wwork = thewin->work.g_w;
X
X         lines_onscrn=hwork/gl_hchar;
X
X        pxy[0] = xwork;
X        pxy[2] = xwork+wwork;
X        pxy[4] = xwork;
X        pxy[6] = xwork+wwork;
X
X    if(upflag==1)
X     {  pxy[1] = ywork+gl_hchar;
X        pxy[3] = ywork+hwork-1;
X        pxy[5] = ywork;
X        pxy[7] = ywork+hwork-1-gl_hchar;
X     }
X    if(dnflag==1)
X     {  pxy[1] = ywork;
X        pxy[3] = ywork+hwork-1-gl_hchar;
X        pxy[5] = ywork+gl_hchar;
X        pxy[7] = ywork+hwork-1;
X     }
X      /* scroll window */
X
X      vro_cpyfm(thewin->graf.handle,S_ONLY,pxy,
X                           &thewin->graf.mfdb,&thewin->graf.mfdb);
X     /* now blank the top or bottom line as appropriate  */
X        pxyc[0] = xwork;
X        pxyc[2] = xwork + wwork - 1;
X
X
X /* if scroll up/dwn then blank out the
X                        last/first line on the screen  */
X     if(upflag == 1)
X      {  pxyc[1] = ywork + (lines_onscrn-1)*gl_hchar + 1;
X         pxyc[3] = ywork + hwork - 1;
X /* set the clip rect to the line to be redrawn */
X         vs_clip(thewin->graf.handle,1,pxyc);
X         vr_recfl(thewin->graf.handle,pxyc);
X        graf_mouse(M_OFF, 0L);
X        one_line2(thewin,thewin->topchar+(thewin->work.g_h/gl_hchar)*16-8);
X        graf_mouse(M_ON, 0L);
X /* the pos passed to one_line2 is topchar+g_h/16*16   */
X      }
```

```
X     if(dnflag == 1)
X      {  pxyc[1] = ywork;
X         pxyc[3] = ywork + gl_hchar;
X /* set the clip rect to the line to be redrawn */
X         vs_clip(thewin->graf.handle,1,pxyc);
X         vr_recfl(thewin->graf.handle,pxyc);
X        graf_mouse(M_OFF, 0L);
X         one_line2(thewin,thewin->topchar);
X        graf_mouse(M_ON, 0L);
X      }
X
X}
SHAR_EOF
chmod 0600 SLIDER.C || echo "restore of SLIDER.C fails"
sed 's/^X//' << 'SHAR_EOF' > WIND.C &&
X#include <osbind.h>
X#include <gemdefs.h>
X#include <obdefs.h>
X
X#include "globals.h"
X
X/*
X    External references for window update procs.
X*/
Xextern one_page();
X
X/*
X    do_window - determines the type of window event and then calls
X         the appropriate function to handle the event.
X*/
Xdo_window(message)
X    int *message;
X{
X    int handle;
X    windowptr thewin;
X
X    handle = message[3];
X
X    graf_mouse(M_OFF, 0L);
X    wind_update(BEG_UPDATE);
X
X    switch (message[0]) {
X         case WM_REDRAW:
X              draw_flag = FALSE;
X              do_update(message);
X         break;
X
X         case WM_NEWTOP:
X         case WM_TOPPED:
X              make_frontwin(findwindowptr(handle));
X         break;
X
X         case WM_MOVED:
```

```
X         case WM_SIZED:
X              do_resize(message);
X              redraw_vslider(handle);
X         break;
X
X         case WM_FULLED:
X              do_fullsize(handle);
X              redraw_vslider(handle);
X         break;
X
X         case WM_CLOSED:
X              thewin = findwindowptr(handle);
X              dispose_window(thewin);
X              dispose_buf(thewin);
X         break;
X
X         case WM_ARROWED:
X              arro_flag = FALSE;
X         case WM_VSLID:
X              slid_flag = FALSE;
X              vhandler(message);
X    }
X
X    wind_update(END_UPDATE);
X    graf_mouse(M_ON, 0L);
X
X}
X
X
X/*
X    do_resize - redraws the window at it's new postion and updates all
X         of the window's position records.
X*/
Xdo_resize(message)
X    int *message;
X{
X    int x, y, w, h;
X    int handle;
X
X    handle   = message[3];
X    x        = (message[4]&~0x7);
X    y        = message[5];
X    w        = ((message[6]&~0x7)+4);
X    h        = message[7];
X
X    /*
X         Make sure that the window doesn't become too small.
X    */
X    if (w < 80) w = 80;
X    if (h < 80) h = 84;
X
X    /*
X         Redraw the window at it's new size.
```

```
X    */
X    wind_set(handle, WF_CURRXYWH, x, y, w, h);
X    wind_get(handle, WF_WORKXYWH, &x, &y, &w, &h);
X
X    {
X         /*
X              Set the Window record data.
X         */
X         windowptr thewin;
X
X         thewin = findwindowptr(handle);
X
X         rect_set(&thewin -> work, x, y, w, h);
X         rect_set(&thewin -> box, x, y, w, h);
X         thewin -> fullsize = FALSE;
X    }
X}
X
X
X/*
X    do_fullsize - draws the window at it's fully defined size.  If the window
X         is at it's full size then this routines restores the window to it's
X         previous size.
X*/
Xdo_fullsize(handle)
X    int handle;
X{
X    register windowptr thewin;
X
X    int x, y, w, h;
X    int d;
X
X    thewin = findwindowptr(handle);
X
X    if (thewin -> fullsize) {
X         /*
X              Back to normal size
X         */
X         wind_calc(WC_WORK, thewin -> kind, thewin -> box,
X              &thewin -> work.g_x, &thewin -> work.g_y,
X              &thewin -> work.g_w, &thewin -> work.g_h);
X
X/* note here that thewin->box, a GRECT structure of 4 ints  */
X/* is passed to wind_calc in place of four ints     jlc     */
X
X         wind_set(handle, WF_CURRXYWH, thewin -> box);
X         thewin -> fullsize = FALSE;
X    } else {
X         /*
X              Draw window at full size;
X         */
X         wind_get(handle, WF_FULLXYWH, &x, &y, &w, &h);
X         wind_set(handle, WF_CURRXYWH, x, y, w, h);
```

```
X         wind_calc(WC_WORK, thewin -> kind, x, y, w, h,
X              &thewin -> work.g_x, &thewin -> work.g_y,
X              &thewin -> work.g_w, &thewin -> work.g_h);
X
X         thewin -> fullsize = TRUE;
X    }
X}
X
X
X/*
X    do_update - Update all of the rectangles affected by the update event.
X*/
Xdo_update(message)
X    int *message;
X{
X    int   thewindow;
X    GRECT r1, therect;
X
X    thewindow = message[3];
X
X    rect_set(&therect, message[4], message[5], message[6], message[7]);
X
X    wind_get(thewindow, WF_FIRSTXYWH, &r1.g_x, &r1.g_y, &r1.g_w, &r1.g_h);
X
X    while (r1.g_w && r1.g_h) {
X         if (rect_intersect(therect, r1, &r1))
X         {
X              setclip(thewindow, &r1);
X              redraw_vslider(thewindow);
X              update_window(thewindow);
X         }
X
X         wind_get(thewindow, WF_NEXTXYWH, &r1.g_x, &r1.g_y, &r1.g_w, &r1.g_h);
X    }
X
X    {
X         int x, y, w, h;
X
X         /*
X              Restore clip rectangle of the desktop rectangle.
X         */
X         wind_get(0, WF_WORKXYWH, &x, &y, &w, &h);
X         rect_set(&r1, x, y, x+w, y+h);
X         vs_clip(phys_handle, 1, &r1);
X    }
X}
X
X
X/*
X    update_window  -  execute the update procedure associated with the window.
X*/
Xupdate_window(windhandle)
X    int           windhandle;
```

```
X{
X    windowptr thewin;
X
X    thewin = findwindowptr(windhandle);
X
X    (*thewin -> updateproc)(thewin,1);
X}
X
X
X/*
X    setclip  -  set the windows clipping rectangle.
X*/
Xsetclip(thewindow, r1)
X    int   thewindow;
X    GRECT *r1;
X{
X    GRECT cliprect;
X    int  grafhandle;
X
X    grafhandle = findwindowptr(thewindow) -> graf.handle;
X
X    rect_set(&cliprect, r1->g_x, r1->g_y, r1->g_x+r1->g_w-1, r1->g_y+r1->g_h-1);
X
X    vs_clip(grafhandle, 1, &cliprect);
X}
X
X
X/*
X                 Window support routines.
X*/
X
X/*
X    findwindowptr  -  find the window record associated with the window
X         handle and return a pointer to that window record.
X*/
Xwindowptr findwindowptr(handle)
X    int handle;
X{
X    register windowptr thewin = firstwindow;
X    int button;
X
X    for (thewin = firstwindow; thewin; thewin = thewin -> next)
X/* tricky for-loop.  When thewin becomes a NULL the loop terminates  */
X         if (thewin -> handle == handle)
X              break;
X
X    if (!thewin) {
X         button = form_alert(1, "[1][ Internal Error: | No window found for
handle ][OK]");
X         shutdown(2);
X    }
X
X    return thewin;
```

```
X}
X
X
X/*
X    new_window  -  create & draw a new window.
X
X    1.)  create the window.
X    2.)  draw the window with the wind_open()
X    3.)  create and setup the window record.
X*/
Xwindowptr new_window(thekind)
X    int           thekind;
X{
X    int           handle, button, dummy, i;
X    int           xdesk, ydesk, wdesk, hdesk;
X    windowptr thewin;
X    static        window_count = 1;
X
X    /*
X         Get the desktop coordinates.
X    */
X    wind_get(0, WF_WORKXYWH, &xdesk, &ydesk, &wdesk, &hdesk);
X
X    /*
X         Create the information for the window.  Max size is the desktop.
X    */
X    handle = wind_create(thekind, xdesk, ydesk, wdesk, hdesk);
X
X    /*
X         Check for error.
X    */
X    if (handle < 0) {
X         button = form_alert(1, "[1][ Sorry! | No more windows available. ][OK]");
X         return NULL;
X    }
X
X    /*
X         Allocate space for window record.
X    */
X    thewin                    = (windowptr) malloc(sizeof(windowrec));
X    if (thewin == NULL)
X    {    button = form_alert(1, "[1][ Sorry! | Out of memory creating window. ]
[OK]");
X         wind_delete(handle); /* delete the window created above  */
X         return NULL;
X    }
X
X
X    /*
X         Initialize window data structure.
X    */
X    thewin -> next            = NULL;
X    thewin -> handle     = handle;
```

```
X    thewin -> kind           = thekind;
X    thewin -> fullsize   = TRUE;
X    thewin -> graf.handle = open_vwork(&thewin -> graf.mfdb);
X    strcpy(thewin->title,"untitled");
X    strcpy(fs_insel,"untitled");
X    wind_set(thewin->handle, WF_NAME, thewin -> title, 0, 0);
X
X /* set text alignment to bottom left  for all windows   */
X   vst_alignment(thewin->graf.handle,0,3,&dummy,&dummy);
X
X    thewin -> updateproc  = one_page;
X
X    thewin -> headptr    = NULL;
X    thewin -> startmark   = 1;
X    thewin -> endmark    = 0;
X    thewin -> markson     = FALSE;
X    thewin -> xcur           = 0;
X    thewin -> ycur           = 0;
X    thewin -> topchar    = 0;
X    thewin -> flen           = 0;
X    thewin -> position   = 0;
X    thewin -> vslidepos   = 0;
X    thewin -> icount     = 0;
X    thewin -> changed    = 0;
X
X    /*
X         Insert into windowlist.
X    */
X    {
X         register windowptr winptr = (windowptr) &firstwindow;
X
X/* firstwindow is a pointer to the first windowrec in the linked list and  */
X/* is initialized to NULL.  Since the first thing in a windowrec is
       */
X/* 'next', the pointer to the next windowrec, then
                        */
X/* '(windowptr)&firstwindow->next' or windptr->next starts at firstwindow */
X/* What a kludge to save a bit of code!  Below is some commented out      */
X/* tested code that is clearer.                                           */
X/*      if(!firstwindow)
X          firstwindow = thewin;
X        else
X        {   winptr = firstwindow;
X             while(winptr -> next)
X             winptr = winptr -> next;
X        winptr -> next = thewin;
X        }
X
X*/
X        while(winptr -> next)
X             winptr = winptr -> next;
X
X        winptr -> next = thewin;
```

```
X    }
X         return(thewin);
X}
X
X
Xopen_window(thewin)
X    windowptr thewin;
X
X{   int           xdesk, ydesk, wdesk, hdesk;
X
X    /*
X         Get the desktop coordinates.
X    */
X    wind_get(0, WF_WORKXYWH, &xdesk, &ydesk, &wdesk, &hdesk);
X
X    /*
X         A little flim-flammery.
X    */
X    graf_growbox(0, 0, 0, 0, xdesk, ydesk, wdesk, hdesk);
X
X    /*
X         Draw the window.
X    */
X    wind_open(thewin->handle, xdesk, ydesk, wdesk, hdesk);
X
X    wind_get(thewin->handle, WF_WORKXYWH, &xdesk, &ydesk, &wdesk, &hdesk);
X    rect_set(&thewin -> work, xdesk, ydesk, wdesk, hdesk);
X
X    wind_get(thewin->handle, WF_CURRXYWH, &xdesk, &ydesk, &wdesk, &hdesk);
X    rect_set(&thewin ->  box, xdesk, ydesk, wdesk, hdesk);
X
X
X    make_frontwin(thewin);
X
X}
X
X/*
X    dispose_window - Closes the window and disposes the storage for
X         the window record.
X*/
Xdispose_window(thewin)
X    windowptr thewin;
X{
X    int x, y, w, h;
X    int handle;
X
X    handle = thewin -> handle;
X
X
X    wind_close(handle);
X
X    wind_get(handle, WF_CURRXYWH, &x, &y, &w, &h);
X
```

```
X    graf_shrinkbox(0, 0, 0, 0, x, y, w, h);
X
X    dispose_win_resources(thewin);
X
X}
X
Xdispose_win_resources(thewin)
X    windowptr thewin;
X{
X        /*
SHAR_EOF
echo "End of part 2, continue with part 3"
echo "3" > s2_seq_.tmp
exit 0
```

```
From 3818236332545002448
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,9c47c3bdf0c87c70,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-11-11 00:03:51 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!cs.utexas.edu!usc!apple!sun-
barr!newstop!sun!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i007: xxed -- Hex/ASCII file editor  part03/03
Keywords: shar, C
Message-ID: <34746@grapevine.uucp>
Date: 11 Nov 89 08:03:51 GMT
Sender: news@grapevine.EBay
Reply-To: charltn@ccu.umanitoba.ca (Jim Charlton)
Lines: 510
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: charltn@ccu.umanitoba.ca (Jim Charlton)
Posting-number: Volume 3, Issue 7
Archive-name: xxed/part03

#!/bin/sh
# this is part 3 of a multipart archive
# do not concatenate these parts, unpack them in order with /bin/sh
# file WIND.C continued
#
CurArch=3
if test ! -r s2_seq_.tmp
then echo "Please unpack part 1 first!"
     exit 1; fi
( read Scheck
  if test "$Scheck" != $CurArch
  then echo "Please unpack part $Scheck next!"
       exit 1;
  else exit 0; fi
) < s2_seq_.tmp || exit 1
```

```
sed 's/^X//' << 'SHAR_EOF' >> WIND.C
X              Remove window record from window list.
X         */
X         register windowptr winptr = (windowptr) &firstwindow;
X         int button;
X
X         wind_delete(thewin->handle);
X
X         while(winptr -> next)
X              if (winptr -> next == thewin)
X                   break;
X              else
X                   winptr = winptr -> next;
X
X         if (winptr -> next)
X              winptr -> next = winptr -> next -> next;
X         else {
X         button = form_alert(1, "[1][ Internal Error: | Window pointer not in
list. ][OK]");
X              shutdown(2);
X               }
X
X         /*
X              Update the front window pointer.
X         */
X         if (!firstwindow)
X              thefrontwin = NULL;
X         else
X              if (winptr == (windowptr) &firstwindow)
X                   make_frontwin(winptr -> next);
X              else
X                   make_frontwin(winptr);
X
X         if(thefrontwin)
X              immed_redraw(thefrontwin);  /* see send.c  */
X
X         /*
X              Close workstation associated with window.
X         */
X         v_clsvwk(thewin -> graf.handle);
X
X         /*
X              Release window storage.
X         */
X         free(thewin);
X
X}
X
X
X/*
X    make_frontwin - Force a window to the front.
X*/
Xmake_frontwin(thewin)
```

```
X    windowptr thewin;
X{
X    wind_set(thewin -> handle, WF_TOP, 0, 0, 0, 0);
X    thefrontwin = thewin;
X}
X
Xrot_wind()
X{   windowptr winptr, thewin;
X
X
X    if(thewin = thefrontwin)
X    {    winptr = firstwindow;
X         while(winptr != thewin)
X              winptr = winptr->next;
X         if((winptr =  winptr->next)!=NULL)
X              make_frontwin(winptr);
X         else
X              make_frontwin(firstwindow);
X    }
X}
X
Xwind_blank(thewin)
X    windowptr thewin;
X{
X    int           grafhandle;
X    GRECT         windrect;
X    int           x, y, w, h;
X
X    grafhandle = thewin -> graf.handle;
X    x = thewin->work.g_x;
X    y = thewin->work.g_y;
X    w = thewin->work.g_w;
X    h = thewin->work.g_h;
X
X    rect_set(&windrect, x, y, x+w, y+h);
X
X    vsf_interior(grafhandle, 1);  /* set for solid fill */
X    vsf_color(grafhandle, 0);    /*  blank to white  */
X
X    v_bar(grafhandle, &windrect);        /* blank the interior */
X}
SHAR_EOF
chmod 0600 WIND.C || echo "restore of WIND.C fails"
sed 's/^X//' << 'SHAR_EOF' > XXED.DEF &&
XMMRCP2.0XSHAR_EOF
chmod 0600 XXED.DEF || echo "restore of XXED.DEF fails"
sed 's/^X//' << 'SHAR_EOF' > XXED.H &&
X#define DIALOG2 2
X#define OK1 8
X#define MENUBAR 0
X#define DESK 3
X#define AA 1
X#define DIALOG4 4
```

```
X#define FILE 4
X#define BB 2
X#define MARK 6
X#define EDIT 5
X#define II 3
X#define SEARCH 7
X#define CC 5
X#define JJ 4
X#define WINDOWS 8
X#define KK 6
X#define INFORM 9
X#define DD 7
X#define EE 9
X#define LL 8
X#define ABOUT 12
X#define MM 10
X#define FF 11
X#define GG 13
X#define NN 12
X#define OO 14
X#define HH 15
X#define PP 16
X#define DRBOX 18
X#define OPEN 22
X#define NEW 21
X#define DRSTR1 19
X#define DREXIT 20
X#define CLOSE 24
X#define SAVEAS 26
X#define SAVE 25
X#define DIALOG3 3
X#define ABBUT 1
X#define DELETE 28
X#define SKBUT 3
X#define RPBUT 2
X#define SPACE 29
X#define PRINT 31
X#define DIALOG5 5
X#define QUIT 33
X#define CUT 35
X#define COPY 36
X#define ERASE 38
X#define PASTE 37
X#define INSERT 40
X#define REPLACE 41
X#define START 43
X#define END 44
X#define CLEAR 45
X#define FIND 47
X#define FORWARD 48
X#define BACK 49
X#define ROTATE 51
X#define POS 54
```

```
X#define FSIZE 53
X#define RVER 1
X#define DIALOG1 1
X#define RALL 5
X#define RONCE 6
X#define SBACK 3
X#define CANCEL1 4
X#define SFORWARD 2
X#define SSTRING 9
X#define DIALOG6 6
X#define RSTRING 11
X#define SSTRBUT 13
X#define SCUTBUF 14
X#define RSTRBUT 17
X#define RCUTBUF 16
Xextern unsigned char XXEDRSC[];
SHAR_EOF
chmod 0600 XXED.H || echo "restore of XXED.H fails"
sed 's/^X//' << 'SHAR_EOF' > XXED.RSC &&
XX:_________________________XXXXXXXXXXXbSHAR_EOF
chmod 0600 XXED.RSC || echo "restore of XXED.RSC fails"
sed 's/^X//' << 'SHAR_EOF' > XXEDRSC.C &&
X/*
X * C version of 'D:\HEXED\XXED.RSC'
X */
X
Xunsigned char XXEDRSC[] = {
X    0,0,4,226,4,142,4,142,4,142,4,142,0,36,4,142,4,142,
X    15,122,0,113,0,7,0,3,0,0,0,0,0,0,0,0,15,150,
X    98,121,0,74,46,32,67,104,97,114,108,116,111,110,0,40,72,101,
X    120,97,100,101,99,105,109,97,108,32,101,100,105,116,111,114,32,118,
X    101,114,115,105,111,110,32,49,46,48,41,0,65,76,76,32,82,73,
X    71,72,84,83,32,82,69,83,69,82,86,69,68,0,0,84,101,108,
X    32,40,50,48,52,41,45,50,54,49,45,55,53,49,50,0,51,48,
X    32,80,114,101,115,99,111,116,32,82,100,44,32,87,105,110,110,105,
X    112,101,103,44,32,67,97,110,97,100,97,0,79,75,0,88,88,69,
X    68,0,32,68,101,115,107,32,0,32,70,105,108,101,32,0,32,69,
X    100,105,116,0,32,77,97,114,107,0,32,83,101,97,114,99,104,32,
X    0,32,87,105,110,100,111,119,115,0,32,32,73,110,102,111,0,32,
X    32,72,101,120,32,69,100,105,116,111,114,0,45,45,45,45,45,45,
X    45,45,45,45,45,45,45,45,45,45,45,45,45,45,0,32,32,68,
X    101,115,107,32,65,99,99,101,115,115,111,114,121,32,49,32,32,0,
X    32,32,68,101,115,107,32,65,99,99,101,115,115,111,114,121,32,50,
X    32,32,0,32,32,68,101,115,107,32,65,99,99,101,115,115,111,114,
X    121,32,51,32,32,0,32,32,68,101,115,107,32,65,99,99,101,115,
X    115,111,114,121,32,52,32,32,0,32,32,68,101,115,107,32,65,99,
X    99,101,115,115,111,114,121,32,53,32,32,0,32,32,68,101,115,107,
X    32,65,99,99,101,115,115,111,114,121,32,54,32,32,0,32,32,32,
X    78,101,119,46,46,46,32,32,0,32,32,32,79,112,101,110,46,46,
X    46,0,45,45,45,45,45,45,45,45,45,45,45,45,45,45,45,0,
X    32,32,32,67,108,111,115,101,32,32,32,0,32,83,32,83,97,118,
X    101,32,32,32,0,32,32,32,83,97,118,101,32,65,115,46,46,46,
X    32,32,0,45,45,45,45,45,45,45,45,45,45,45,45,45,45,45,
```

```
X    0,32,32,32,68,101,108,101,116,101,46,46,46,0,32,32,32,68,
X    105,115,107,32,83,112,97,99,101,0,45,45,45,45,45,45,45,45,
X    45,45,45,45,45,45,45,0,32,32,32,80,114,105,110,116,0,45,
X    45,45,45,45,45,45,45,45,45,45,45,45,45,0,32,81,32,
X    81,117,105,116,32,32,32,32,32,32,32,0,32,88,32,67,117,116,
X    0,32,67,32,67,111,112,121,32,32,0,32,86,32,80,97,115,116,
X    101,32,0,32,69,32,69,114,97,115,101,32,0,45,45,45,45,45,
X    45,45,45,45,45,45,0,32,32,32,73,110,115,101,114,116,0,32,
X    32,32,82,101,112,108,97,99,101,0,32,73,32,77,97,114,107,32,
X    83,116,97,114,116,0,32,79,32,77,97,114,107,32,69,110,100,32,
X    0,32,80,32,67,108,101,97,114,32,77,97,114,107,115,32,0,32,
X    70,32,70,105,110,100,0,32,65,32,70,111,114,119,97,114,100,0,
X    32,66,32,66,97,99,107,0,32,82,111,116,97,116,101,32,119,105,
X    110,100,111,119,115,0,32,70,105,108,101,32,115,105,122,101,58,0,
X    32,80,111,115,105,116,105,111,110,58,0,65,0,66,0,73,0,74,
X    0,67,0,75,0,68,0,76,0,69,0,77,0,70,0,78,0,71,
X    0,79,0,72,0,80,0,83,69,76,69,67,84,32,68,82,73,86,
X    69,0,0,69,88,73,84,0,65,66,79,82,84,0,82,69,80,76,
X    65,67,69,0,83,75,73,80,0,83,65,86,73,78,71,32,70,73,
X    76,69,0,82,101,112,108,97,99,101,32,65,108,108,32,119,105,116,
X    104,32,86,101,114,105,102,121,0,82,101,112,108,97,99,101,32,65,
X    108,108,0,82,101,112,108,97,99,101,32,79,110,99,101,0,67,65,
X    78,67,69,76,0,83,101,97,114,99,104,32,66,65,67,75,87,65,
X    82,68,83,0,83,101,97,114,99,104,32,70,79,82,87,65,82,68,
X    0,83,101,97,114,99,104,32,97,110,100,32,82,101,112,108,97,99,
X    101,32,68,105,97,108,111,103,32,66,111,120,0,95,95,95,95,95,
X    95,0,83,101,97,114,99,104,32,32,83,116,114,105,110,103,58,95,
X    95,95,95,95,95,95,95,95,95,95,95,95,95,95,95,95,95,95,
X    95,95,95,95,95,0,88,88,88,88,88,88,88,88,88,88,88,88,
X    88,88,88,88,88,88,88,88,88,88,88,88,0,95,95,95,95,95,
X    0,82,101,112,108,97,99,101,32,83,116,114,105,110,103,58,95,95,
X    95,95,95,95,95,95,95,95,95,95,95,95,95,95,95,95,95,95,
X    95,95,95,95,0,88,88,88,88,88,88,88,88,88,88,88,88,88,
X    88,88,88,88,88,88,88,88,88,88,88,88,0,83,101,97,114,99,104,
X    32,115,116,114,105,110,103,0,67,117,116,32,98,117,102,102,101,114,
X    0,67,117,116,32,98,117,102,102,101,114,0,82,101,112,108,97,99,
X    101,32,115,116,114,105,110,103,0,83,101,97,114,99,104,32,102,111,
X    114,58,0,82,101,112,108,97,99,101,32,119,105,116,104,58,0,68,
X    69,76,69,84,73,78,71,32,70,73,76,69,0,0,0,0,0,84,
X    0,0,0,0,0,0,0,104,0,5,0,6,0,0,17,128,0,0,
X    0,255,0,20,0,1,0,0,3,163,0,0,3,170,0,0,3,210,
X    0,3,0,6,0,0,17,128,0,0,0,255,0,7,0,40,0,0,
X    3,235,0,0,3,241,0,0,4,25,0,3,0,6,0,0,17,128,
X    0,0,0,255,0,6,0,40,255,255,0,1,0,7,0,20,0,0,
X    0,16,0,2,17,0,0,0,0,0,0,58,0,16,0,2,255,255,
X    255,255,0,28,0,0,0,0,0,0,0,36,0,26,0,6,0,2,
X    0,1,0,3,255,255,255,255,0,28,0,0,0,0,0,0,0,39,
X    0,22,0,8,0,11,0,1,0,4,255,255,255,255,0,28,0,0,
X    0,0,0,0,0,51,0,12,0,4,0,32,0,1,0,5,255,255,
X    255,255,0,21,0,0,0,0,0,0,4,142,0,20,0,11,0,15,
X    0,1,0,6,255,255,255,255,0,28,0,0,0,0,0,0,0,105,
X    0,18,0,10,0,18,0,1,0,8,255,255,255,255,0,28,0,0,
X    0,0,0,0,0,124,0,12,0,9,0,31,0,1,0,0,255,255,
```

```
X    255,255,0,28,0,32,0,0,0,0,0,159,0,25,0,2,0,4,
X    0,1,0,7,255,255,255,255,0,26,0,7,0,0,0,0,0,156,
X    0,23,0,13,0,8,0,1,255,255,0,1,0,10,0,25,0,0,
X    0,0,0,0,0,0,0,0,0,0,0,80,0,25,0,10,0,2,
X    0,2,0,20,0,0,0,0,0,0,17,0,0,0,0,0,0,80,
X    2,1,0,1,0,3,0,9,0,25,0,0,0,0,0,0,0,0,
X    0,2,0,0,0,51,3,1,0,4,255,255,255,255,0,32,0,0,
X    0,0,0,0,0,164,0,0,0,0,0,6,3,1,0,5,255,255,
X    255,255,0,32,0,0,0,0,0,0,0,171,0,6,0,0,0,6,
X    3,1,0,6,255,255,255,255,0,32,0,0,0,0,0,0,0,178,
X    0,12,0,0,0,6,3,1,0,7,255,255,255,255,0,32,0,0,
X    0,0,0,0,0,184,0,18,0,0,0,6,3,1,0,8,255,255,
X    255,255,0,32,0,0,0,0,0,0,0,190,0,24,0,0,0,9,
X    3,1,0,9,255,255,255,255,0,32,0,0,0,0,0,0,0,199,
X    0,33,0,0,0,9,3,1,0,2,255,255,255,255,0,32,0,0,
X    0,0,0,0,0,208,0,42,0,0,0,9,3,1,0,0,0,11,
X    0,52,0,25,0,0,0,0,0,0,0,0,0,3,1,0,80,
X    0,19,0,20,0,12,0,19,0,20,0,0,0,0,0,255,17,0,
X    0,2,0,0,0,20,0,8,0,13,255,255,255,255,0,28,0,0,
X    0,0,0,0,0,215,0,0,0,0,0,20,0,1,0,14,255,255,
X    255,255,0,28,0,0,0,8,0,0,0,228,0,0,0,1,0,20,
X    0,1,0,15,255,255,255,255,0,28,0,0,0,0,0,0,0,249,
X    0,0,0,2,0,20,0,1,0,16,255,255,255,255,0,28,0,0,
X    0,0,0,0,1,14,0,0,0,3,0,20,0,1,0,17,255,255,
X    255,255,0,28,0,0,0,0,0,0,1,35,0,0,0,4,0,20,
X    0,1,0,18,255,255,255,255,0,28,0,0,0,0,0,0,1,56,
X    0,0,0,5,0,20,0,1,0,19,255,255,255,255,0,28,0,0,
X    0,0,0,0,1,77,0,0,0,6,0,20,0,1,0,11,255,255,
X    255,255,0,28,0,0,0,0,0,0,1,98,0,0,0,7,0,20,
X    0,1,0,34,0,21,0,33,0,20,0,0,0,0,0,255,17,0,
X    0,8,0,0,0,15,0,13,0,22,255,255,255,255,0,28,0,0,
X    0,0,0,0,1,119,0,0,0,0,0,15,0,1,0,23,255,255,
X    255,255,0,28,0,0,0,0,0,0,1,131,0,0,0,1,0,15,
X    0,1,0,24,255,255,255,255,0,28,0,0,0,8,0,0,1,142,
X    0,0,0,2,0,15,0,1,0,25,255,255,255,255,0,28,0,0,
X    0,0,0,0,1,158,0,0,0,3,0,15,0,1,0,26,255,255,
X    255,255,0,28,0,0,0,0,0,0,1,170,0,0,0,4,0,15,
X    0,1,0,27,255,255,255,255,0,28,0,0,0,0,0,0,1,181,
X    0,0,0,5,0,15,0,1,0,28,255,255,255,255,0,28,0,0,
X    0,8,0,0,1,197,0,0,0,6,0,15,0,1,0,29,255,255,
X    255,255,0,28,0,0,0,0,0,0,1,213,0,0,0,7,0,15,
X    0,1,0,30,255,255,255,255,0,28,0,0,0,0,0,0,1,226,
X    0,0,0,8,0,15,0,1,0,31,255,255,255,255,0,28,0,0,
X    0,8,0,0,1,240,0,0,0,9,0,15,0,1,0,32,255,255,
X    255,255,0,28,0,0,0,0,0,0,2,0,0,0,0,10,0,15,
X    0,1,0,33,255,255,255,255,0,28,0,0,0,8,0,0,2,9,
X    0,0,0,11,0,15,0,1,0,20,255,255,255,255,0,28,0,0,
X    0,0,0,0,2,25,0,0,0,12,0,15,0,1,0,42,0,35,
X    0,41,0,20,0,0,0,0,0,255,17,0,0,14,0,0,0,11,
X    0,7,0,36,255,255,255,255,0,28,0,0,0,8,0,0,2,40,
X    0,0,0,0,0,11,0,1,0,37,255,255,255,255,0,28,0,0,
X    0,8,0,0,2,47,0,0,0,1,0,11,0,1,0,38,255,255,
X    255,255,0,28,0,0,0,8,0,0,2,57,0,0,0,2,0,11,
```

```
X    0,1,0,39,255,255,255,255,0,28,0,0,0,8,0,0,2,67,
X    0,0,0,3,0,11,0,1,0,40,255,255,255,255,0,28,0,0,
X    0,8,0,0,2,77,0,0,0,4,0,11,0,1,0,41,255,255,
X    255,255,0,28,0,0,0,0,0,0,2,89,0,0,0,5,0,11,
X    0,1,0,34,255,255,255,255,0,28,0,0,0,4,0,0,2,99,
X    0,0,0,6,0,11,0,1,0,46,0,43,0,45,0,20,0,0,
X    0,0,0,255,17,0,0,20,0,0,0,15,0,3,0,44,255,255,
X    255,255,0,28,0,0,0,0,0,0,2,110,0,0,0,0,0,15,
X    0,1,0,45,255,255,255,255,0,28,0,0,0,0,0,0,2,124,
X    0,0,0,1,0,15,0,1,0,42,255,255,255,255,0,28,0,0,
X    0,0,0,0,2,137,0,0,0,2,0,15,0,1,0,50,0,47,
X    0,49,0,20,0,0,0,0,0,255,17,0,0,26,0,0,0,11,
X    0,3,0,48,255,255,255,255,0,28,0,0,0,0,0,0,2,153,
X    0,0,0,0,0,10,0,1,0,49,255,255,255,255,0,28,0,0,
X    0,0,0,0,2,161,0,0,0,1,0,11,0,1,0,46,255,255,
X    255,255,0,28,0,0,0,0,0,0,2,172,0,0,0,2,0,11,
X    0,1,0,52,0,51,0,51,0,20,0,0,0,0,0,255,17,0,
X    0,35,0,0,0,16,0,1,0,50,255,255,255,255,0,28,0,0,
X    0,0,0,0,2,180,0,0,0,0,0,16,0,1,0,10,0,53,
X    0,54,0,20,0,0,0,0,0,255,17,0,0,44,0,0,0,20,
X    0,2,0,54,255,255,255,255,0,28,0,8,0,0,0,0,2,196,
X    0,0,0,0,0,20,0,1,0,52,255,255,255,255,0,28,0,40,
X    0,0,0,0,2,208,0,0,0,1,0,20,0,1,255,255,0,1,
X    0,20,0,20,0,0,0,16,0,2,17,0,0,0,0,0,0,44,
X    0,12,0,2,255,255,255,255,0,26,0,21,0,8,0,0,2,219,
X    0,3,0,2,0,3,0,1,0,3,255,255,255,255,0,26,0,21,
X    0,8,0,0,2,221,0,8,0,2,0,3,0,1,0,4,255,255,
X    255,255,0,26,0,21,0,8,0,0,2,223,0,3,0,4,0,3,
X    0,1,0,5,255,255,255,255,0,26,0,21,0,8,0,0,2,225,
X    0,8,0,4,0,3,0,1,0,6,255,255,255,255,0,26,0,21,
X    0,8,0,0,2,227,0,13,0,2,0,3,0,1,0,7,255,255,
X    255,255,0,26,0,21,0,8,0,0,2,229,0,13,0,4,0,3,
X    0,1,0,8,255,255,255,255,0,26,0,21,0,8,0,0,2,231,
X    0,18,0,2,0,3,0,1,0,9,255,255,255,255,0,26,0,21,
X    0,8,0,0,2,233,0,18,0,4,0,3,0,1,0,10,255,255,
X    255,255,0,26,0,21,0,8,0,0,2,235,0,23,0,2,0,3,
X    0,1,0,11,255,255,255,255,0,26,0,21,0,8,0,0,2,237,
X    0,23,0,4,0,3,0,1,0,12,255,255,255,255,0,26,0,21,
X    0,8,0,0,2,239,0,28,0,2,0,3,0,1,0,13,255,255,
X    255,255,0,26,0,21,0,8,0,0,2,241,0,28,0,4,0,3,
X    0,1,0,14,255,255,255,255,0,26,0,21,0,8,0,0,2,243,
X    0,33,0,2,0,3,0,1,0,15,255,255,255,255,0,26,0,21,
X    0,8,0,0,2,245,0,33,0,4,0,3,0,1,0,16,255,255,
X    255,255,0,26,0,21,0,8,0,0,2,247,0,38,0,2,0,3,
X    0,1,0,17,255,255,255,255,0,26,0,21,0,8,0,0,2,249,
X    0,38,0,4,0,3,0,1,0,18,255,255,255,255,0,28,0,0,
X    0,0,0,0,2,251,0,15,0,6,0,12,0,1,0,20,0,19,
X    0,19,0,20,0,0,0,0,0,255,1,0,0,2,0,7,0,40,
X    0,2,0,18,255,255,255,255,0,28,0,0,0,0,0,0,3,8,
X    0,1,0,1,0,37,0,1,0,0,255,255,255,255,0,26,0,39,
X    0,0,0,0,3,9,0,17,0,10,0,8,0,1,255,255,0,1,
X    0,3,0,20,0,0,0,16,0,2,17,0,0,0,0,0,0,31,
X    0,3,0,2,255,255,255,255,0,26,0,21,0,0,0,0,3,14,
```

```
X    0,21,0,1,0,8,0,1,0,3,255,255,255,255,0,26,0,23,
X    0,0,0,0,3,20,0,1,0,1,0,8,0,1,0,0,255,255,
X    255,255,0,26,0,53,0,0,0,0,3,28,0,11,0,1,0,8,
X    0,1,255,255,0,1,0,1,0,20,0,0,0,0,0,2,17,0,
X    0,0,0,0,0,22,0,1,0,0,255,255,255,255,0,28,0,32,
X    0,0,0,0,3,33,0,6,0,0,0,11,0,1,255,255,0,1,
X    0,19,0,20,0,0,0,16,0,2,17,0,0,0,0,0,0,59,
X    0,17,0,5,255,255,255,255,0,26,0,5,0,0,0,0,3,45,
X    0,30,0,15,0,25,0,1,0,7,255,255,255,255,0,26,0,7,
X    0,0,0,0,3,118,0,3,0,11,0,21,0,1,0,2,255,255,
X    255,255,0,26,0,5,0,0,0,0,3,101,0,3,0,13,0,21,
X    0,1,0,3,255,255,255,255,0,26,0,5,0,0,0,0,3,94,
X    0,3,0,15,0,21,0,1,0,6,255,255,255,255,0,26,0,5,
X    0,0,0,0,3,69,0,30,0,13,0,25,0,1,0,4,255,255,
X    255,255,0,26,0,5,0,0,0,0,3,81,0,30,0,11,0,25,
X    0,1,0,8,255,255,255,255,0,28,0,0,0,0,0,0,3,133,
X    0,15,0,1,0,29,0,1,0,10,0,9,0,9,0,20,0,0,
X    0,0,0,255,17,0,0,12,0,3,0,39,0,1,0,8,255,255,
X    255,255,0,29,0,8,0,0,0,0,4,170,0,0,0,0,0,39,
X    0,1,0,12,0,11,0,11,0,20,0,0,0,0,0,255,17,0,
X    0,12,0,5,0,39,0,1,0,10,255,255,255,255,0,29,0,8,
X    0,0,0,0,4,198,0,0,0,0,0,39,0,1,0,15,0,13,
X    0,14,0,20,0,0,0,0,0,255,1,0,0,25,0,7,0,29,
X    0,1,0,14,255,255,255,255,0,26,0,17,0,0,0,0,4,50,
X    0,1,0,0,0,15,0,1,0,12,255,255,255,255,0,26,0,17,
X    0,0,0,0,4,64,0,17,0,0,0,11,0,1,0,18,0,16,
X    0,17,0,20,0,0,0,0,0,255,1,0,0,25,0,9,0,29,
X    0,1,0,17,255,255,255,255,0,26,0,17,0,0,0,0,4,75,
X    0,17,0,0,0,11,0,1,0,15,255,255,255,255,0,26,0,17,
X    0,0,0,0,4,86,0,1,0,0,0,15,0,1,0,19,255,255,
X    255,255,0,28,0,0,0,0,0,0,4,101,0,8,0,7,0,11,
X    0,1,0,0,255,255,255,255,0,28,0,32,0,0,0,0,4,113,
X    0,8,0,9,0,13,0,1,255,255,0,1,0,1,0,20,0,0,
X    0,0,0,2,17,0,0,0,0,0,0,22,0,1,0,0,255,255,
X    255,255,0,28,0,32,0,0,0,0,4,127,0,4,0,0,0,13,
X    0,1,0,0,5,186,0,0,13,106,0,0,4,226,0,0,12,218,
X    0,0,10,226,0,0,13,58,0,0,15,74
X};
SHAR_EOF
chmod 0600 XXEDRSC.C || echo "restore of XXEDRSC.C fails"
sed 's/^X//' << 'SHAR_EOF' > XXEDSRC.DOC &&
X                       XXED -   HEXADECIMAL/ASCII  EDITOR
X
X                  Written and distributed by  Jim Charlton
X                       charltn@ccm.umanitoba.ca
X                  or   charltn@ccu.umanitoba.ca
X                       30 Prescot Road
X                       Winnipeg, Manitoba
X                       CANADA   R3T 3X5
X                       Tel (204)-261-7267 voice
X                            (204)-261-7512 bbs (STadel)
X
X
```

X   XXED copyright Jim Charlton 1989. XXED is freely distributable
Xprovided that this file is distributed with the source and executable
Xcode. No-one may charge a fee for the executable or source code, nor for
Xdistribution of this code.
X
X   Version 1.0 is the first release of this program and all reports of
Xproblems, bugs or suggestions for improvements should be forwarded to the
Xaddress given above.
X
X   No garantees, written or implied are made as to the suitability of
Xthis program for any particular purpose. Anyone who uses it, does so at
Xtheir own risk.
X
X   The source code was compiled using Megamax Laser C version 2.0. The
Xprogram was tested primarily on a 1040ST (Nov 1985 ROMS), monochrome and
Xhas not been extensively tested in colour, although it should work in
Xcolour.
X
X   The file FNCTLST.DOC contains a list of all of the functions defined
Xin the source code with very brief descriptions of what they do. While
Xthe source code is not extensively documented, it should be readable by
Xan experience C programmer. The basic program structure is modelled after
Xthe example code that comes with the Laser C compiler. Each file has its
Xown window and workspace. A linked list is used to keep track of the
Xopen windows. Each window has a file buffer which is a linked list of
Xblocks. Using linked lists makes memory management easier and speeds
Xup inserts and deletes.
X
X   The following very brief instructions should suffice to enable you to
Xuse the program.
X
XFeatures:
X   1.   Can create and edit files of any format.
X   2.   Can insert or replace bytes using hexadecimal or ascii input.
X   3.   Displays simultaneously in hex and ascii.
X   4.   Can cut, copy, paste and erase.
X   5.   Can search or search/replace on hex or ascii strings.
X   6.   Full GEM interface. Open up to 7 files in individual windows.
X
XInstructions:
X   Start the program and inspect the menu entries. Under file you can
Xopen an existing file or start editing a new file. When you open a file,
Xa copy of the file is made in ram which can be edited. This program is
Xnot a disk editor, and does not change the copy of the file on the disk
Xuntil you specifically save the file. Having opened a file, you can edit
Xin hexadecimal or ascii mode. The input mode is switched by clicking the
Xmouse in the hex or ascii display area. You can either insert or replace
Xbytes. The insert/replace mode is toggled with the insert key or the edit
Xmenu. Selected text can be marked using the cursor and the mark menu or
Xby dragging the mouse in either the hex or ascii display area (Only the
Xhex display is marked). The marked area can be copied or cut to a
Xcutbuffer, or erased using the edit menu. Text which has been copied or
Xcut to the cutbuffer can be pasted into the file at the cursor position

```
Xusing the paste command in the edit menu.  You can search forwards and
Xbackwards for ascii strings from the search menu.  You can also search and
Xreplace from the search menu.  You can use the cutbuffer as either the search
Xor replace pattern.  In this way you can also search or search/replace on
Xhex strings.  The info menu provides the length of the file and the
Xposition of the cursor in the file.  The print option under the file menu
Xwill print the file in the active window or the marked text (if any) in the
Xactive window.  Pressing any key during printing will abort printing.
X
X
SHAR_EOF
chmod 0600 XXEDSRC.DOC || echo "restore of XXEDSRC.DOC fails"
rm -f s2_seq_.tmp
echo "You have unpacked the last part"
exit 0


From 7524527462917172143
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,55b682a721198abc,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-11-14 20:41:41 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!wuarchive!texbell!texsun!
newstop!sun!sunfedcomm!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i008: nroff -- Partial clone of UNIX "nroff" formatter  part01/03
Keywords: shar, C
Message-ID: <34764@grapevine.uucp>
Date: 15 Nov 89 04:41:41 GMT
Sender: daemon@sunfedcomm
Reply-To: rosenkra@hall.cray.com (Bill Rosenkranz)
Lines: 1369
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: rosenkra@hall.cray.com (Bill Rosenkranz)
Posting-number: Volume 3, Issue 8
Archive-name: nroff/part01

this is source for a partially complete nroff(1) for atariST TOS. it should
port to minix-ST (it runs on 4BSD).

part01/03

-bill
rosenkra@hall.cray.com
rosenkra@boston.cray.com

---------cut here-------------------cut here-----------------cut here-------
#!/bin/sh
# This is a shell archive.  Remove anything before this line,
# then unpack it by saving it in a file and typing "sh file".
#
```

```
# Wrapped by hall!rosenkra on Sun Nov 12 16:35:26 CST 1989
# Contents:  README MANIFEST Makefile Makefile.bsd nroff.1 tmac.an nroff.h
#    version.h

echo x - README
sed 's/^@//' > "README" <<'@//E*O*F README//'
nroff (TOS) v0.90 ALPHA 11/12/89 wjr
-----------------------------------

This is a preliminary release of a version of nroff for atariST. It was
written under TOS, USG, and 4BSD but should port easily to Minix ST as
well. I doubt it will be easy to port to Minix PC unless the static arrays
are changed to be malloc'd.

Much of this works. The things that don't are all the hard things (natch):
diversions, traps, conditionals, etc. Font switching, underlining,
etc. are limited to reverse video on the screen, though stdout will
contain underlines as "_^H" for printers.

There is a crude man package included (tmac.an) which is not right
but needs diversions to become right. It should handle most man tasks,
though, except .IP and similar commands.

For porting info, look in nroff.h and the makefile. I have to fix
my ctime(3) so as implemented, it is slightly incorrect. The standard
dri/alcyon libraries are limited so I have my own extensions. Only
ctime(3) and getenv(3) are used, as I recall. If you use dLibs or MWC
you should be in good shape (SOMEDAY I'll switch to dLibs). Minix should
also have no problems. As it stands, I have a >300k stack set up (plenty
of memory on a Mega4 *8^) and this should be fine for a 1040ST. If it
is not, and you don't/can't recompile, consider dumping some of your DA's.
You could also binary edit the executable, looking for a long in the bss
segment which defines the stack size for the startup (I use a hacked up
gemstart.s). Look for 0x00055555L long near the end (NB: 0x55 is "U").
I think is is at offset $a9e0 in the file.

If you do hack away, I would appreciate you sending me the changes
so I can keep this centralized. Eventually, this may be good enough
for FSF (GNUroff?).

-Bill Rosenkranz
net: rosenkra@hall.cray.com, rosenkra@boston.cray.com
CIS: 71460,17
GENIE:   W.ROSENKRANZ

@//E*O*F README//
chmod u=rw,g=r,o=r README

echo x - MANIFEST
sed 's/^@//' > "MANIFEST" <<'@//E*O*F MANIFEST//'
-rw-r--r--  1 rosenkra      906 Nov 12 15:40 MANIFEST
-rw-r--r--  1 rosenkra     1195 Nov 12 15:40 Makefile
-rw-r--r--  1 rosenkra     1268 Nov 12 16:28 Makefile.bsd
```

```
-rw-r--r--  1 rosenkra     1841 Nov 12 15:40 README
-rw-r--r--  1 rosenkra    16799 Nov 12 15:40 command.c
-rw-r--r--  1 rosenkra    17014 Nov 12 15:53 escape.c
-rw-r--r--  1 rosenkra     3444 Nov 12 15:40 io.c
-rw-r--r--  1 rosenkra    10846 Nov 12 15:40 low.c
-rw-r--r--  1 rosenkra     9095 Nov 12 15:40 macros.c
-rw-r--r--  1 rosenkra    15924 Nov 12 16:16 main.c
-rw-r--r--  1 rosenkra    20809 Nov 12 16:05 nroff.1
-rw-r--r--  1 rosenkra    13272 Nov 12 15:40 nroff.h
-rw-r--r--  1 rosenkra     4920 Nov 12 15:40 strings.c
-rw-r--r--  1 rosenkra    11452 Nov 12 15:40 text.c
-rw-r--r--  1 rosenkra      547 Nov 12 15:40 tmac.an
-rw-r--r--  1 rosenkra      292 Nov 12 15:40 version.h
@//E*O*F MANIFEST//
chmod u=rw,g=r,o=r MANIFEST

echo x - Makefile
sed 's/^@//' > "Makefile" <<'@//E*O*F Makefile//'
# makefile for nroff (TOS)
#
CFLAGS        = -new -Dalcyon
# LDFLAGS for no symbols (-s) and to specify _STKSIZ in crt0.o (-stksiz)
LDFLAGS       = -new -s -stksiz
SRCS     = main.c command.c text.c io.c macros.c strings.c escape.c low.c
OBJS     = main.o command.o text.o io.o macros.o strings.o escape.o low.o
# LIBS is to get my own getenv, ctime, etc.
LIBS     = -llibc
TARGET        = nroff.ttp
MANPAGE       = nroff.1
MAN_MACRO= tmac.an
#MS_MACRO= tmac.s
#ME_MACRO= tmac.e
# set these as per your system (for install)
TMACDIR       = c:\lib\tmac
BINDIR        = c:\bin
MANDIR        = c:\man\man1


# main target
#
$(TARGET):    $(OBJS)
         $(CC) $(LDFLAGS) -o $(TARGET) $(OBJS) $(LIBS)


# install target
#
install: $(TARGET)
         cp $(TARGET) $(BINDIR)\$(TARGET)
         cp $(MANPAGE) $(MANDIR)\$(MANPAGE)
         cp $(MAN_MACRO) $(TMACDIR)\$(MAN_MACRO)
#        cp $(MS_MACRO) $(TMACDIR)\$(MS_MACRO)
#        cp $(ME_MACRO) $(TMACDIR)\$(ME_MACRO)
```

```
clean:
	rm $(OBJS)

clobber:
	rm $(OBJS)
	rm $(TARGET)

# dependencies
#
nroff.h: version.h
	touch nroff.h

main.o:	main.c nroff.h
command.o:	command.c nroff.h
text.o:	text.c nroff.h
io.o:	io.c nroff.h
macros.o: macros.c nroff.h
strings.o:	strings.c nroff.h
escape.o: escape.c nroff.h
low.o:	low.c nroff.h

@//E*O*F Makefile//
chmod u=rw,g=r,o=r Makefile

echo x - Makefile.bsd
sed 's/^@//' > "Makefile.bsd" <<'@//E*O*F Makefile.bsd//'
# makefile for nroff (BSD)
#
CFLAGS	= -O
LDFLAGS	=
SRCS	= main.c command.c text.c io.c macros.c strings.c escape.c low.c
OBJS	= main.o command.o text.o io.o macros.o strings.o escape.o low.o
LIBS	=
TARGET	= nroff
MANPAGE	= nroff.1
MAN_MACRO = tmac.an
#MS_MACRO = tmac.s
#ME_MACRO = tmac.e
# set these as per your system (for install)
TMACDIR	= /usr/local/lib/tmac
BINDIR	= /usr/local/bin
MANDIR	= /usr/local/man/man1


# main target
#
$(TARGET):	$(OBJS)
	$(CC) $(LDFLAGS) -o $(TARGET) $(OBJS) $(LIBS)


# install target
#
install: $(TARGET)
```

```
	cp $(TARGET) $(BINDIR)/$(TARGET)
	cp $(MANPAGE) $(MANDIR)/$(MANPAGE)
	cp $(MAN_MACRO) $(TMACDIR)/$(MAN_MACRO)
#	cp $(MS_MACRO) $(TMACDIR)/$(MS_MACRO)
#	cp $(ME_MACRO) $(TMACDIR)/$(ME_MACRO)

clean:
	/bin/rm -f $(OBJS)

clobber:
	/bin/rm -f $(OBJS)
	/bin/rm -f $(TARGET)

shar:
	shar README MANIFEST Makefile Makefile.bsd nroff.1 tmac.an \
	    *.h >part01
	shar main.c command.c text.c io.c >part02
	shar escape.c low.c macros.c strings.c >part03

# dependencies
#
nroff.h: version.h
	touch nroff.h

main.o:	main.c nroff.h
command.o:	command.c nroff.h
text.o:	text.c nroff.h
io.o:	io.c nroff.h
macros.o: macros.c nroff.h
strings.o:	strings.c nroff.h
escape.o: escape.c nroff.h
low.o:	low.c nroff.h

@//E*O*F Makefile.bsd//
chmod u=rw,g=r,o=r Makefile.bsd

echo x - nroff.1
sed 's/^@//' > "nroff.1" <<'@//E*O*F nroff.1//'
@.\" a macro to do a dangling tag starting with a "."
@.de XT
@.br
@.sp
@.ne 3
@.ti -6
\&\fI$0\fR $1 $2 $3 $4 $5 $6 $7 $8 $9
@.br
@..
@.TH NROFF 1 "Commands Manual" "Version 1.0" "Sept 1989"
@.SH NAME
nroff - text processor

@.SH SYNOPSIS
\fInroff\fR [\fIoptions\fR] \fIfile\fR [\fI...\fR]
```

```
@.SH DESCRIPTION
\fINroff\fR is a text processor and formatter based on the design
provided in "Software Tools" by Kernighan and Plauger.
It has been modified to closely resemble the Unix(tm) nroff command.
The text and commands found in the \fIfile(s)\fR
are processed to generate formatted text.
Note that one (and only one) of the files can be "-" which reads
input from stdin at that point.
The output always goes to stdout which can be redirected by the shell.
@.\" Using the command line option \fI-l\fR will cause the output to
@.\" be sent to the printer instead.
The \fI-o\fR option lets you redirect error output to the specified
file rather than stderr.
Debugging information always goes to the file "nroff.dbg" and is
generally used only for program development.

@.SH OPTIONS
The following command line options are available:
@.nf

\fI-a\fR            Changes font (not implemented).

\fI-b\fR            Allows backspaces to appear in the output
              text  when  underlining  or overstriking.
              This  has  the  same  effect  as  the \fI\.bs\fR
              command with a non-zero argument.

\fI-d\fR            Set debug mode.

\fI-h\fR            Hold screen before desktop (TOS only).

\fI-l\fR            Send output to printer (not implemented).

\fI-m<name>\fR      Process  macro  file  tmac.<name>.   Thus
              -man would cause the file  tmac.an  to be
              loaded. Note that files processed in this
              way should contain only macro definitions,
              no  immediate output  should be generated
              from this file (see ENVIRONMENT).

\fI-o\fR \fIfile\fR       Set error log file (default is stderr).

\fI-po<n>\fR        Shift output right n spaces (like \fI.po\fR).

\fI-pn<n>\fR        Initial page number (like \fI.pn\fR).

\fI-v\fR            Prints the version information to stdout.

\fI+<n>\fR          Causes output to start with page n.

\fI-<n>\fR          Causes output to stop after page n.
```

```
\fI-\fR             Input from stdin.

@.fi

@.SH ENVIRONMENT
Nroff recognizes the following environment variables from the shell:
@.br
@.in +6

@.XT TMACDIR
An alternate directory to find the files tmac.* ("." for example).
The default is c:\\lib\\tmac under TOS.
@.br
@.in -6

@.SH FILES
@.nf
c:\\lib\\tmac\\tmac.*      predefined macros (see ENVIRONMENT)
nroff.dbg               debugging output
stderr                  default error output stream
stdout                  output stream

@.fi

@.SH COMMANDS
Commands typically are distinguished by a period in column one
of the input
followed by a two character abbreviation for the command funtion.
The abbreviation may then be followed by an optional numeric or
character argument.
The numeric argument may be an absolute value such as setting
the right margin to a particular column, or the argument may be
preceded by a plus sign or a minus sign to indicate that the
parameter should be modified relative to a previous setting.
The following commands are recognized (those marked "extension"
are requests added to the basic set):
@.br
@.in +6

@.XT .bo (extension)
causes the following lines of text to appear in boldface.
The optional argument specifies the number of lines to be typed in boldface.
Boldface and underlining are mutually exclusive features.
The appearance of a boldface command will cause any underlining to cease.

@.XT .bp (extension)
causes succeeding text to appear at the top of a new page.
The optional argument specifies the page number for the new page.
The initial value is one and the default value is one more than
the previous page number.

@.XT .br
causes succeeding text to start on a new line at the current left margin.
```

There is no numeric argument for this command.

@.XT .bs (extension)
enables or disables the appearance of backspaces in the output text.
Underlining and boldface options are implemented by inserting
character - backspace - character combinations into the output buffer.
This is fine for devices which properly recognize the backspace character.
Some printers, however, do not recognize backspaces, so the option is
provided to overprint one line buffer with another.
The first line buffer is terminated with just a carriage return
rather than the carriage return - linefeed combination.
A zero argument or no argument to the backspace command removes
backspaces from the output.
A non-zero argument leaves them in the output.
The default is to remove backspaces.

@.XT .cc
changes the \fInroff\fR command character to that specified by the
character argument.
If no argument is provided, the default is a period.

@.XT .ce
causes the next line of text to appear centered on the output.
The optional argument specifies if more than one line is to be centered.

@.XT .c2
changes the \fInroff\fR no break character to that specified by the
character argument.
If no argument is provided, the default is a single quote.

@.XT .de
causes all text and commands following to be used to define a macro.
The definition is terminated by a \fI\.en\fR command or the
default \fI\.\.\fR terminator.
The first two characters of the argument following the \fI\.de\fR
command become the name of the new command.
It should be noted that upper and lower case arguments are considered different.
Thus, the commands \fI\.PP\fR and \fI\.pp\fR could define two different macros.
Care should be exercised since existing commands may be redefined.
@.PP
A macro may contain up to ten arguments.
In the macro definition, the placement of arguments is designated by the
two character sequences, $0, $1, ... $9.
When the macro is invoked, each argument of the macro command line is
substituted for its corresponding designator in the expansion.
The first argument of the macro command is substituted for the $0
in the expansion, the second argument for the $1, and so forth.
Arguments are typically strings which do not contain blanks or tabs.
If an argument is to contain blanks, then it should be surrounded by
either single or double quotes.

@.XT .cu
causes the next line(s) of text to be continuously underlined.

Unlike the underline command (see \fI\.ul\fR) which underlines only
alphanumerics, continuous underlining underlines all printable characters.
The optional argument specifies the number of lines of text to underlined.
Any normal underlining or boldface commands currently in effect will be
terminated.

@.XT .ec
changes the \fInroff\fR escape character to that specified by the
character argument.
If no argument is provided, the default is a backslash.

@.XT .ef (extension)
specifies the text for the footer on even numbered pages.
The format is the same as for the footer command (see \fI\.fo\fR).

@.XT .eh (extension)
specifies the text for the header on even numbered pages.
The format is the same as for the footer command (see \fI\.fo\fR).

@.XT .en (extension)
designates the end of a macro definition.

@.XT .fi
causes the input text to be rearranged or filled to obtain the maximum
word count possible between the previously set left and right margins.
No argument is expected.

@.XT .fl
causes the output buffer to be flushed immediately.

@.XT .fo (extension)
specifies text to be used for a footer.
The footer text contains three strings seperated by a delimiter character.
The first non-blank character following the command is designated
as the delimiter.
The first text string is left justified to the current indentation
value (specified by \fI\.in\fR).
The second string is centered between the current indentation value
and the current right margin value (specified by \fI\.rm\fR).
The third string is right justified to the current right margin value.
The absence of footer text will result in the footer being printed as
one blank line.
The presence of the page number character (set by \fI\.pc\fR) in the footer
text results in the current page number being inserted at that position.
Multiple occurrances of the page number character are allowed.

@.XT .ft
changes the current font.
The choices are R (Times Roman), I (Times Italic), B (Times Bold),
S (math special), and P used to request the previous font.
P resets the next previous font to be the one just changed, amounting to a swap.

@.XT .he (extension)

specifies text to be used for a header.
The format is the same as for the footer (see \fI\.fo\fR).

@.XT .in
indents the left margin to the column value specified by the argument.
The default left margin is set to zero.

@.XT .ju (extension)
causes blanks to be inserted between words in a line of
output in order to align or justify the right margin.
The default is to justify.

@.XT .ll
sets the current line length. The default is eighty.

@.XT .ls
sets the line spacing to the value specified by the argument.
The default is for single spacing.

@.XT .m1 (extension)
specifies the number of lines in the header margin.
This is the space from the physical top of page to and including
the header text.
A value of zero causes the header to not be printed.
A value of one causes the header to appear at the physical top of page.
Larger argument values cause the appropriate number of blank
lines to appear before the header is printed.

@.XT .m2 (extension)
specifies the number of blank lines to be printed between
the header line and the first line of the processed text.

@.XT .m3 (extension)
specifies the number of blank lines to be printed between
the last line of processed text and the footer line.

@.XT .m4 (extension)
specifies the number of lines in the footer margin.
This command affects the footer the same way the \fI\.m1\fR
command affects the header.

@.XT .ne
specifies a number of lines which should not be broken across a page boundary.
If the number of lines remaining on a page is less than the
value needed, then a new output page is started.

@.XT .nf
specifies that succeeding text should be printed without
rearrangement, or with no fill.
No argument is expected.

@.XT .nj (extension)
specifies that no attempt should be made to align or justify the right margin.

No argument is expected.

@.XT .nr
causes the value of a number register to be set or modified.
A total of twenty-six number registers are available designated
\\na through \\nz (either upper or lower case is allowed).
When the sequence \\nc is imbedded in the text, the current value
of number register c replaces the sequence, thus, such things as
paragraph numbering can be accomplished with relative ease.

@.XT .of (extension)
specifies the text for the footer on odd numbered pages.
The format is the same as the footer command (see \fI\.fo\fR).

@.XT .oh (extension)
specifies the text for the header on odd numbered pages.
The format is the same as the footer command (see \fI\.fo\fR).

@.XT .pc
specifies the page number character to be used in headers and footers.
The occurrance of this character in the header or footer text
results in the current page number being printed.
The default for this character is the percent sign (%).

@.XT .pl
specifies the page length or the number of lines per output page.
The default is sixty-six.

@.XT .po
specifies a page offset value.
This allows the formatted text to be shifted to the right by
the number of spaces specified.
This feature may also be invoked by a switch on the command line.

@.XT .pn
changes the page number of the current page and all
subsequent pages to its argument. If no argument is given,
the command is ignored.

@.XT .so
causes input to be retrieved from the file specified
by the command's character string argument.
The contents of the new file are inserted into the output
stream until an EOF is detected.
Processing of the original file is then resumed.
Command nesting is allowed.

@.XT .sp
specifies a number of blank lines to be output before
printing the next line of text.

@.XT .ti
temporarily alters the indentation or left margin value for a single

```
succeeding input line.

@.XT .tl
specifies text to be used for a page title.
The format is the same as for the header (see \fI\.he\fR).

@.XT .ul
causes the next line(s) of text to be underlined.
Unlike the \fI\.cu\fR command, this command causes only alphanumerics
to be underlined, skipping punctuation and white space.
Underline and boldface are mutually exclusive.
@.br
@.in -6


@.SH ESCAPE SEQUENCES
Escape sequences are used to access special characters (such as Greek
letters) which may be outside the normal printable ASCII character set.
The are also used to toggle certain actions such as font selection.
@.PP
@.ne 5
The escape sequences include:
@.sp
@.eo
@.nf
     \\          backslash character
     \"          beginning of comment
     \(xx        special character named xx
     \fc         font change (c = R,I,B,S,P)
     \nx         interpolate number register x
@.fi
@.PP
@.ne 5
The special characters include:
@.sp
@.nf
     \(co        copyrite
     \(rg        registered
     \(tm        trademark
     \(12        1/2
     \(14        1/4
     \(p2        exponent 2
     \(p3        exponent 3
     \(pn        exponent n
     \(aa        acute
     \(ga        grave
     \(de        degree
     \(dg        dagger
     \(ct        cent
     \(bu        bullet
     \(pp        paragraph
     \(^g        ring bell
     \(ua        up arrow
```

```
     \(da        dn arrow
     \(->        rt arrow
     \(<-        lf arrow
     \(di        divide
     \(sr        sq root
     \(==        ==
     \(>=        >=
     \(<=        <=
     \(+-        +-
     \(~=        ~=
     \(ap        approx
     \(no        not
     \(mo        memeber
     \(ca        intersect
     \(cu        union
     \(*a        alpha
     \(*b        beta
     \(*g        gamma
     \(*d        delta
     \(*s        sigma
     \(*p        pi
     \(*m        mu
@.fi
@.ec
@.PP
@.SH REQUEST SUMMARY
The following table summarizes the nroff requests:
@.nf
@.ne 8
@.cc +

Request Form    Initial  Default  Notes   Explanation
--------------  -------  -------  ------- ----------------------
Font and Character Size Control

@.ps \(+-N         10pt     prev     E       point size
@.ss N           12/36em  ignored  E       space-char size
@.cs F N M       off      -        P       constant space mode
@.bd F N         off      -        P       embolden font F
@.bd S F N       off      -        P       embolden special font
@.ft F           Roman    prev     E       change to font F
@.fp N F         R,I,B,S  ignored  -       font F in position N

+ne 4
Page Control

@.pl \(+-N         11in     11in     v       page length
@.bp \(+-N         N=1      -        B,v     eject page
@.pn \(+-N         N=1      ignored  -       next page number N
@.po \(+-N         0        prev     v       page offset
@.ne N           -        N=1V     D,v     need N vertical space
@.mk R           none     intern   D       mark current V in R
@.rt \(+-N         none     intern   D,v     return (up) to mark
```

```
+ne 4
Text Filling, Adjusting, and Centering

@.br             -        -        B       break
@.fi             fill     -        B,E     fill mode
@.nf             fill     -        B,E     no fill or adjust
@.ad c           adj,both adjust   E       adjust output, mode c
@.na             adjust   -        E       no adjust
@.ce N           off      N=1      B,E     center N lines

+ne 4
Vertical Spacing

@.vs N           1/6in    prev     E,p     vert. baseline space
@.ls N           N=1      prev     E       output N-1 Vs
@.sp N           -        N=1V     B,v     space vertical
@.sv N           -        N=1V     v       save vertical dist N
@.os             -        -        D       output saved vert dist
@.ns             space    -        D       no-space mode on
@.rs             -        -        D       restore spacing

+ne 4
Line Length and Indenting

@.ll \(+-N          6.5i     prev     E,m     line length
@.in \(+-N          N=0      prev     B,E,m   indent
@.ti \(+-N          -        ignored  B,E,m   temporary indent

+ne 4
Macros, Strings, Diversions, and Position Traps

@.de xx yy       -        .yy=..   -       define macro xx
@.am xx yy       -        .yy=..   -       append to macro xx
@.ds xx str      -        ignored  -       define string xx
@.as xx str      -        ignored  -       append to string xx
@.rm xx          -        ignored  -       remove macro, string
@.rn xx yy       -        ignored  -       rename macro, string
@.di xx          -        end      D       divert output to xx
@.da xx          -        end      D       divert and append to xx
@.wh N xx        -        -        v       set location trap
@.ch xx N        -        -        v       change trap location
@.dt N xx        -        off      D,v     set diversion trap
@.it N xx        -        off      E       set line count trap
@.em xx          none     none     -       end macro is xx

+ne 4
Number Registers

@.nr R \(+-N M              -        u       define and set num reg
@.af R c         arabic   -        -       assign format to reg
@.rr R           -        -        -       remove register
```

```
+ne 4
Tabs, Leaders, and Fields

@.ta Nt ...      0.8      none     E,m     tab set
@.tc c           none     none     E       tab repeat char
@.lc c           .        none     E       leader repeat char
@.fc a b         off      off      -       set field delim, pad

+ne 4
I/O Conventions and Character Translation

@.ec c           \        \        -       set escape char
@.eo             on       -        -       turn off escape mech
@.lg N           -        on       -       ligature mode if N>0
@.ul N           off      N=1      E       underline N lines
@.cu N           off      N=1      E       cont. underline
@.uf F           Italic   Italic   -       underline font
@.cc c           .        .        E       set control char
@.c2 c           '        '        E       set nobreak control char
@.tr abcd...     none     -        O       translate a-b, etc

+ne 4
Hyphenation

@.nh             hyphen   -        E       no hyphenation
@.hy N           hyphen   hyphen   E       hyphenate, N=mode
@.hc c           \%       \%       E       hyphanation char
@.hw word1 ...   none     ignored  -       exception words

+ne 4
Three-part Titles

@.tl 'l'c'r'             -        -       three-part title
@.pc c           %        off      -       page number char
@.lt \(+-N          6.5in    prev     E,m     length of title

+ne 4
Output Line Numbering

@.nm \(+-N M S I            off      E       number mode
@.nn N           -        N=1      E       do not number N lines

+ne 4
Conditional Acceptance of Input

@.if c ...               -        -       if condition c true
@.if !c ...              -        -       if condition c false
@.if N ...               -        u       if expression N>0
@.if !N ...              -        u       if expression N<=0
@.if 's1's2' ...         -        -       if strings identical
@.if !'s1's2' ...        -        -       if strings differ
@.ie c ...               -        u       if portion of if/else
@.ie !c ...              -        u       if portion of if/else
```

```
@.ie N ...                -        u       if portion of if/else
@.ie !N ...               -        u       if portion of if/else
@.ie 's1's2' ...          -        u       if portion of if/else
@.ie !'s1's2' ...         -        u       if portion of if/else
@.el ...                  -        u       else portion of if/else

+ne 4
Environment Switching

@.ev N           N=0      prev     -       environment switched

+ne 4
Insertions From the Standard Input

@.rd prompt      -        BEL      -       read insertion
@.ex             -        -        -       exit nroff

+ne 4
Input/Output File Switching

@.so filename             -        -       switch source file
@.nx filename             EOF      -       next file
@.pi program              -        -       pipe output to program

+ne 4
Miscellaneous

@.mc c N         -        off      E,m     set margin char
@.tm string      -        newline  -       terminal message
@.ig yy          -        .yy=..   -       ignore till call to yy
@.pm t           -        all      -       print macro names
@.fl             -        -        B       flush output buffer

+ne 4
Notes

B       causes a break
D       mode or parameters associated with current diversion level
E       relevant parameters are a part of the current environment
O       must stay in effect until logical output
P       mode must be still or again in effect at time of physical output
v,p,m,u default scale indicators

+cc
@.fi

@.SH AUTHOR
@.nf
Adapted for atariST/TOS by Bill Rosenkranz 11/89

net: rosenkra@hall.cray.com
CIS: 71460,17
GENIE:   W.ROSENKRANZ
```

Original author:

Stephen L. Browning
5723 North Parker Avenue
Indianapolis, Indiana 46220

History:

- Originally written in BDS C
- Adapted for standard C by W. N. Paul
- Heavily hacked up to conform to "real" nroff by Bill Rosenkranz

```
@.fi
@//E*O*F nroff.1//
chmod u=rw,g=r,o=r nroff.1

echo x - tmac.an
sed 's/^@//' > "tmac.an" <<'@//E*O*F tmac.an//'
@.\" -man package for nro
@.\" not quite unix, but working...
@.\"
@.de TH                 \" main page heading
@.m1 1
@.in 0
@.lt 65
@.ll 65
@.tl |$0 ($1)|$2|$0 ($1)|
@.fo |$3|$4|%|
@.in 5
@.ll 60
@..
@.de pp                 \" start a new indented paragraph
@.sp 1
@.ti +5
@..
@.de PP                 \" start a new unindented paragraph
@.sp 1
@..
@.de SH                 \" section heading
@.sp 1
@.ne 3
@.ti -5
@.bo
$0 $1 $2 $3 $4 $5 $6 $7 $8 $9
@.br
@..
@.de SS                 \" subsection heading, same indent
@.sp 1
@.ne 3
@.bo
$0 $1 $2 $3 $4 $5 $6 $7 $8 $9
@.br
```

```
@.sp 1
@..
@.de I                  \" italic text
\fB$0 $1 $2 $3 $4 $5 $6 $7 $8 $9\fR
@..
@//E*O*F tmac.an//
chmod u=rw,g=r,o=r tmac.an

echo x - nroff.h
sed 's/^@//' > "nroff.h" <<'@//E*O*F nroff.h//'
#ifndef NRO_H
#define NRO_H

#ifdef alcyon
#define ALCYON                  /* for gemdos version, alcyon C */
                        /* otherwise minix/unix */
#endif

/*
 *   nroff.h - stuff for nroff
 *
 *   adapted for atariST/TOS by Bill Rosenkranz 10/89
 *   net: rosenkra@hall.cray.com
 *   CIS: 71460,17
 *   GENIE:    W.ROSENKRANZ
 *
 *   things to look for here:
 *   1) TMAC definition for default macro package lib
 *   2) configuration sizes (see _STKSIZ below if alcyon/dri)
 *   3) libc should have getenv() and ctime()
 *   4) look in version.h for *printer file name (included below)
 *
 *   original author:
 *
 *   Stephen L. Browning
 *   5723 North Parker Avenue
 *   Indianapolis, Indiana 46220
 *
 *   history:
 *
 *   - Originally written in BDS C;
 *   - Adapted for standard C by W. N. Paul
 *   - Heavily hacked up to conform to "real" nroff by Bill Rosenkranz
 */

#include <ctype.h>

/*
 *   default prefix of macro files. files will be of the form "tmac.an"
 *   (for -man), "tmac.s" (for -ms), "tmac.e" (for -me), etc. first
 *   checks environment for TMACDIR which would be path (e.g. "c:\lib\tmac"
 *   or ".", no trailing slash char!).
 */
```

```
#ifdef ALCYON
#define TMACFULL   "c:\\lib\\tmac\\tmac."
#define TMACPRE         "\\tmac."
#else /*unix/minix???*/
#define TMACFULL   "/usr/lib/tmac/tmac."
#define TMACPRE         "/tmac."
#endif

/*
 *   command codes...
 */
#define MACRO      0    /* macro definition */
#define BP         1    /* begin page */
#define BR         2    /* break */
#define CE         3    /* center */
#define FI         4    /* fill  */
#define FO         5    /* footer */
#define HE         6    /* header */
#define IN         7    /* indent */
#define LS         8    /* line spacing    */
#define NF         9    /* no fill */
#define PL         10   /* page length */
#define RM         11   /* remove macro */
#define SP         12   /* line space */
#define TI         13   /* temp indent */
#define UL         14   /* underline */
#define JU         15   /* justify */
#define NJ         16   /* no justify */
#define M1         17   /* top margin */
#define M2         18   /* second top margin */
#define M3         19   /* first bottom margin */
#define M4         20   /* bottom-most margin */
#define BS         21   /* allow/disallow '\b' in output */
#define NE         22   /* need n lines */
#define PC         23   /* page number character (%) */
#define CC         24   /* control character (.) */
#define PO         25   /* page offset */
#define BO         26   /* bold face */
#define EH         27   /* header for even numbered pages */
#define OH         28   /* header for odd numbered pages */
#define EF         29   /* footer for even numbered pages */
#define OF         30   /* footer for odd numbered pages */
#define SO         31   /* source file */
#define CU         32   /* continuous underline*/
#define DE         33   /* define macro    */
#define EN         34   /* end macro definition*/
#define NR         35   /* set number register */
#define EC         36   /* escape character (\) */
#define FT         37   /* font change (R,B,I,S,P) */
#define EO         38   /* turn escape parsing off */
#define LL         39   /* line length (same as RM) */
#define FL         40   /* flush output NOW */
#define PN         41   /* page number for next page */
```

```
#define RR         42   /* remove register */
#define C2         43   /* nobreak char */
/*   these 3 not done yet... */
#  define TR       44   /* translate character */
#  define LT       45   /* length of title */
#  define FC       46   /* field delimeter */
#define TL         47   /* like HE */
#define AF         48   /* assign format to nr */
#define AD         49   /* adjust line */
#define NA         50   /* no adjust */
#define DS         51   /* define string */
#define PM         52   /* print macro names */
#define IF         53   /* if */
#define IE         54   /* if/else */
#define EL         55   /* else */
/*   these aren't done yet either... */
#  define PS       56   /* point size */
#  define SS       57   /* space char size */
#  define CS       58   /* constant char space */
#  define BD       59   /* bold font */
#  define FP       60   /* font position */
#  define MK       61   /* mark vertical place */
#  define RT       62   /* return to marked vert place */
#  define VS       63   /* vertical baseline spacing */
#  define SV       64   /* save vertical distance */
#  define OS       65   /* output saved vertical distance */
#  define NS       66   /* no-space mode */
#  define RS       67   /* restore spacing mode */
#  define AM       68   /* append to macro */
#  define AS       69   /* append to string */
#  define RN       70   /* rename */
#  define DI       71   /* divert to macro */
#  define DA       72   /* divert/append to macro */
#  define WH       73   /* set location trap */
#  define CH       74   /* change trap location */
#  define DT       75   /* set diversion trap */
#  define IT       76   /* set input line trap */
#  define EM       77   /* end macro */
#  define TA       78   /* tab settings */
#  define TC       79   /* tab repetition char */
#  define LC       80   /* leader repetition char */
#  define LG       81   /* ligature mode */
#  define UF       82   /* underline font */
#  define NH       83   /* no hyphenation */
#  define HY       84   /* hyphenate */
#  define HC       85   /* hyphenation indication char */
#  define HW       86   /* hyphenation exception words */
#  define NM       87   /* number mode */
#  define NN       88   /* no number next lines */
#  define EV       89   /* environment switch */
#  define RD       90   /* read insertion */
#  define EX       91   /* exit */
#  define NX       92   /* next file */
```

```
#  define PI       93   /* pipe to program */
#  define MC       94   /* set margin char */
#  define TM       95   /* print to terminal */
#  define IG       96   /* ignore */


#define COMMENT        1000 /* comment (.\") */

#define UNKNOWN        -1

/*
 *   MAXLINE is set to a value slightly larger than twice the longest
 *   expected input line. Because of the way underlining is handled, the
 *   input line which is to be underlined, can almost triple in length.
 *   Unlike normal underlining and boldfacing, continuous underlining
 *   affects all characters in the buffer, and represents the worst case
 *   condition.  If the distance between the left margin and the right
 *   margin is greater than about 65 characters, and continuous underlining
 *   is in effect, there is a high probability of buffer overflow.
 */
#define MAXLINE        200
#define PAGELEN        66
#define PAGEWIDTH  80
#define HUGE       256
#define INFINITE   32000
#define LEFT       0    /* indecies into hdr margin lim arrays*/
#define RIGHT      1
#define Nfiles         4    /* nesting depth for input files */

/*
 *   The following parameters may be defined elsewhere so undef/def
 */
#undef min
#undef max
#undef YES
#define YES        1
#undef NO
#define NO         0
#undef ERR
#define ERR        -1
#define EOS        '\0'
#undef FALSE
#define FALSE          0
#undef TRUE
#define TRUE       !FALSE
#undef OK
#define OK         !ERR

#define ADJ_OFF        0
#define ADJ_LEFT   1
#define ADJ_RIGHT  2
#define ADJ_CENTER 3
#define ADJ_BOTH   4
```

```
/*
 *   The parameter values selected for macro definitions are somewhat
 *   arbitrary.  MACBUF is the storage area for both macro names and
 *   definitions.  Since macro processing is handled by pushing back
 *   the expansion into the input buffer, the longest possible expansion
 *   would be MAXLINE characters.  Allowing for argument expansion,
 *   MXMLEN was chosen slightly less than MAXLINE. It is assumed that
 *   most macro definitions will not exceed 20 characters, hence MXMDEF
 *   of 150.
 */

#define MXMDEF          150  /* max no. of macro definitions */
#define MACBUF          32000    /* macro definition buffer size */
#define MXMLEN          250  /* max length of each macro def */
#define MNLEN      10   /* max length of macro name */
#define MAXREGS         100  /* max number of registers (2-char) */

struct macros
{
     char   *mnames[MXMDEF];/* table of ptrs to macro names */
     int  lastp;        /* index to last mname  */
     char   *emb;       /* next char avail in macro defn buf */
     char     mb[MACBUF];   /* table of macro definitions */
     char   *ppb;       /* pointer into push back buffer */
     char     pbb[MAXLINE]; /* push back buffer */
};


#define RF_READ         0x0001    /* register flags */
#define RF_WRITE   0x0002

struct regs
{
     char rname[4];/* 2-char register name */
     int  rauto;        /* autoincrement value */
     int  rval;         /* current value of the register */
     int  rflag;        /* register flags */
     char rfmt;         /* register format (1,a,A,i,I,...) */
};



/*
 *   control parameters for nro
 */
struct docctl
{
     int  fill;         /* fill if YES, init = YES */
     int  dofnt;        /* handle font change, init = YES */
     int  lsval;        /* current line spacing, init = 1 */
     int  inval;        /* current indent, >= 0, init = 0 */
```

```
    int  rmval;        /* current right margin, init = 60 */
    int  llval;        /* current line length, init = 60 */
    int  ltval;        /* current title length, init = 60 */
    int  tival;        /* current temp indent, init = 0 */
    int  ceval;        /* number of lines to center, init = 0 */
    int  ulval;        /* number of lines to underline, init = 0 */
    int  cuval;        /* no lines to continuously uline, init = 0 */
    int  juval;        /* justify if YES, init = YES */
    int  adjval;       /* adjust type, init = ADJ_BOTH */
    int  boval;        /* number of lines to bold face, init = 0 */
    int  bsflg;        /* can output contain '\b', init = FALSE */
    int  prflg;        /* print on or off, init = TRUE */
    int  sprdir;       /* direction for spread(), init = 0 */
    int  flevel;       /* nesting depth for source cmd, init = 0 */
    int  lastfnt;  /* previous used font */
    int  thisfnt;  /* current font, init = 1 (1=R,2=I,3=B,4=S) */
    int  escon;        /* whether esc parsing is on, init = YES */
    int  nr[26];       /* number registers */
    int  nrauto[26];   /* number registers auto increment */
    char nrfmt[26];    /* number registers formats, init = '1' */
                 /* input code how printed */
                 /* 1     '1'  1,2,3,... */
                 /* a     'a'  a,b,c,...,aa,bb,cc,... */
                 /* A     'A'  A,B,C,...,AA,BB,CC,... */
                 /* i     'i'  i,ii,iii,iv,v... */
                 /* I     'I'  I,II,III,IV,V... */
                 /* 01     2   01,02,03,... */
                 /* 001    3   001,002,003,... */
                 /* 0..1   8   00000001,00000002,... */
    char pgchr;        /* page number character, init = '%' */
    char cmdchr;       /* command character, init = '.' */
    char escchr;       /* escape char, init = '\' */
    char nobrchr;  /* nobreak char, init = '\'' */
};


/*
 *   output buffer control parameters
 */
struct cout
{
    int  outp;         /* next avail char pos in outbuf, init = 0 */
    int  outw;         /* width of text currently in buffer */
    int  outwds;       /* number of words in buffer, init = 0 */
    int  lpr;     /* output to printer, init = FALSE */
    int  outesc;       /* number of escape char on this line */
    char outbuf[MAXLINE];/* output of filled text */
};


/*
 *   page control parameters for nroff
 */
```

```
struct page
{
     int  curpag;       /* current output page number, init =0 */
     int  newpag;       /* next output page number, init = 1 */
     int  lineno;       /* next line to be printed, init = 0 */
     int  plval;        /* page length in lines, init = 66 */
     int  m1val;        /* margin before and including header */
     int  m2val;        /* margin after header */
     int  m3val;        /* margin after last text line */
     int  m4val;        /* bottom margin, including footer */
     int  bottom;       /* last live line on page
                        = plval - m3val - m4val*/
     int  offset;       /* page offset from left, init = 0 */
     int  frstpg;       /* first page to print, init = 0 */
     int  lastpg;       /* last page to print, init = 30000 */
     int  ehlim[2];/* left/right margins for headers/footers */
     int  ohlim[2];/* init = 0 and PAGEWIDTH */
     int  eflim[2];
     int  oflim[2];
     char ehead[MAXLINE];    /* top of page title, init = '\n' */
     char ohead[MAXLINE];
     char efoot[MAXLINE];    /* bottom of page title, init = '\n' */
     char ofoot[MAXLINE];
};



/*
 *   forward refs from libc
 */
char   *getenv ();
char   *ctime ();      /* NOTE: my ctime is slightly different */

/*
 *   forward refs from nroff
 */
char   *getmac ();
char   *getstr ();
char   *skipwd ();
char   *skipbl ();
char   *getfield ();

int  comand ();
int  comtyp ();
int  gettl ();
int  getval ();
int  set ();
int  expesc ();
int  specialchar ();
int  fontchange ();
int  getlin ();
int  ngetc ();
int  pbstr ();
```

```
int  putbak ();
int  prchar ();
int  put ();
int  putlin ();
int  atod ();
int  robrk ();
int  ctod ();
int  space ();
int  getwrd ();
int  countesc ();
int  itoda ();
int  itoROMAN ();
int  itoroman ();
int  itoLETTER ();
int  itoletter ();
int  min ();
int  max ();
int  defmac ();
int  colmac ();
int  putmac ();
int  maceval ();
int  main ();
int  usage ();
int  init ();
int  pswitch ();
int  profile ();
int  text ();
int  bold ();
int  center ();
int  expand ();
int  justcntr ();
int  justleft ();
int  justrite ();
int  leadbl ();
int  pfoot ();
int  phead ();
int  puttl ();
int  putwrd ();
int  skip ();
int  spread ();
int  strkovr ();
int  underl ();
int  width ();

int  findreg ();
int  set_ireg ();



/*
 *   globals. define NRO_MAIN in main.c to define globals there. else
 *   you get extern.
 */
```

```
#ifdef NRO_MAIN

struct docctl     dc;
struct page       pg;
struct cout       co;
struct macros     mac;
struct regs       rg[MAXREGS];
FILE            *pout;
FILE            *err_stream;
FILE            *dbg_stream;
FILE            *sofile[Nfiles+1];
int           hold_screen;
int           debugging;
char            *dbgfile = "nroff.dbg";
#ifdef ALCYON
char            *printer = "prn:";   /* this WON'T work!!! */
#else
char            *printer = "/dev/lp";     /* this probably won't */
#endif

#include "version.h"            /* for myname and version */

#ifdef ALCYON
/*
 *   this SHOULD be big enough for most needs. only used by startup
 *   code (gemstart.o or crt0.o)
 */
long          _STKSIZ = 0x00055555L;  /* about 350000 */
#endif


#else /*NRO_MAIN*/

extern struct docctl   dc;
extern struct page pg;
extern struct cout co;
extern struct macros   mac;
extern struct regs rg[MAXREGS];
extern FILE          *pout;
extern FILE          *err_stream;
extern FILE          *dbg_stream;
extern FILE          *sofile[Nfiles+1];
extern int         hold_screen;
extern int         debugging;
extern char          *dbgfile;
extern char          *printer;
extern char          *myname;
extern char          *version;

#endif /*NRO_MAIN*/

#endif /*NRO_H*/
```

```
@//E*O*F nroff.h//
chmod u=rw,g=r,o=r nroff.h

echo x - version.h
sed 's/^@//' > "version.h" <<'@//E*O*F version.h//'
#ifndef VERSION_H
#define VERSION_H

/*
 *   to get around no valid argv[0]...
 */
char        *myname  = "nroff";

#ifdef ALCYON
char        *version = "nroff (TOS) v0.90 ALPHA 11/12/89 wjr";
#else
char        *version = "nroff (Minix) v0.90 ALPHA 11/12/89 wjr";
#endif


#endif /*VERSION_H*/
@//E*O*F version.h//
chmod u=rw,g=r,o=r version.h

exit 0


From -485278946920913566
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,6ee706c13e00085b,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-11-14 20:41:54 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!wuarchive!texbell!texsun!
newstop!sun!sunfedcomm!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i009: nroff -- Partial clone of UNIX "nroff" formatter  part02/03
Keywords: shar, C
Message-ID: <34765@grapevine.uucp>
Date: 15 Nov 89 04:41:54 GMT
Sender: daemon@sunfedcomm
Reply-To: rosenkra@hall.cray.com (Bill Rosenkranz)
Lines: 2857
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: rosenkra@hall.cray.com (Bill Rosenkranz)
Posting-number: Volume 3, Issue 9
Archive-name: nroff/part02


part02/03
```

-bill
rosenkra@hall.cray.com

------------cut here----------------cut here---------------------------------
#!/bin/sh
This is a shell archive. Remove anything before this line,
then unpack it by saving it in a file and typing "sh file".
#
Wrapped by hall!rosenkra on Sun Nov 12 16:35:33 CST 1989
Contents: main.c command.c text.c io.c

echo x - main.c
sed 's/^@//' > "main.c" <<'@//E*O*F main.c//'
/*
 *   main.c - main for nroff word processor
 *
 *   similar to Unix(tm) nroff or RSX-11M RNO. adaptation of text processor
 *   given in "Software Tools", Kernighan and Plauger.
 *
 *   adapted for atariST/TOS by Bill Rosenkranz 11/89
 *   net: rosenkra@hall.cray.com
 *   CIS: 71460,17
 *   GENIE: W.ROSENKRANZ
 *
 *   original author:
 *
 *   Stephen L. Browning
 *   5723 North Parker Avenue
 *   Indianapolis, Indiana 46220
 *
 *   history:
 *
 *   - Originally written in BDS C;
 *   - Adapted for standard C by W. N. Paul
 *   - Heavily hacked up to conform to "real" nroff by Bill Rosenkranz
 */

#define NRO_MAIN   /* to define globals in nro.h */

#include <stdio.h>
#ifdef ATARIST
#include <sys\types.h>
#include <sys\time.h>
#else
#include <sys/types.h>
#include <sys/time.h>
#endif
#include "nroff.h"

main (argc, argv)
int   argc;
char   *argv[];

```
{
     register int  i;
     int      swflg;
     int      ifp = 0;


     swflg       = FALSE;
     hold_screen = FALSE;
     debugging   = FALSE;
     pout        = stdout;
     err_stream  = stderr;
     dbg_stream  = stderr;


     /*
      *   initialize structures (defaults)
      */
     init ();


     /*
      *   parse cmdline flags
      */
     for (i = 1; i < argc; ++i)
     {
          if (*argv[i] == '-' || *argv[i] == '+')
          {
               if (pswitch (argv[i], argv[i+1], &swflg) == ERR)
                    err_exit (-1);
          }
     }


     /*
      *   loop on files
      */
     for (i = 1; i < argc; ++i)
     {
          if (*argv[i] != '-' && *argv[i] != '+')
          {
               /*
                *   open this file...
                */
               if ((sofile[0] = fopen (argv[i], "r")) == NULL)
               {
                    fprintf (err_stream,
                         "***%s: unable to open file %s\n",
                         myname, argv[i]);
                    err_exit (-1);
               }
               else
               {
                    /*
```

```
                    *   do it for this file...
                    */
                   ifp = 1;
                   profile ();
                   fclose (sofile[0]);
              }
         }
         else if (*argv[i] == '-' && *(argv[i]+1) == 0)
         {
              /*
               *   - means read stdin
               */
              sofile[0] = stdin;
              ifp = 1;
              profile ();
         }

    }


    /*
     *   if no files, usage
     */
    if ((ifp == 0 && swflg == FALSE) || argc <= 1)
    {
         usage ();

         err_exit (-1);
    }

    /*
     *   if not going to stdout (-l)
     */
    if (pout != stdout)
    {
         fflush (pout);
         fclose (pout);
    }

    err_exit (0);
}




/*------------------------------*/
/*   usage              */
/*------------------------------*/
usage ()
{
    /*
     *   note: -l not documented
     */
```

```
    fprintf (stderr, "Usage: %s [options] file [...]\n", myname);
    fprintf (stderr, "Options:\n");
    fprintf (stderr, "-a              font changes\n");
    fprintf (stderr, "-b              backspace\n");
    fprintf (stderr, "-h              hold screen\n");
/*  fprintf (stderr, "-l              output to printer\n");*/
    fprintf (stderr, "-m<name>        macro file (e.g. -man)\n");
    fprintf (stderr, "-o file         error log file\n");
    fprintf (stderr, "-po<n>          page offset\n");
    fprintf (stderr, "-pn<n>          initial page number\n");
    fprintf (stderr, "-v              version\n");
    fprintf (stderr, "+<n>            first page to do\n");
    fprintf (stderr, "-<n>            last page to do\n");
    fprintf (stderr, "-               use stdin\n");
}



/*------------------------------*/
/*  init          */
/*------------------------------*/
init ()
{

/*
 *  initialize parameters for nro word processor
 */

    register long i;
#ifdef ALCYON
    time_t        dum = 0;
#else
    extern long   time ();
    time_t        dum = time (0);
#endif
    char       *ctim;

    dc.fill    = YES;
    dc.dofnt   = YES;
    dc.lsval   = 1;
    dc.inval   = 0;
    dc.rmval   = PAGEWIDTH - 1;
    dc.llval   = PAGEWIDTH - 1;
    dc.ltval   = PAGEWIDTH - 1;
    dc.tival   = 0;
    dc.ceval   = 0;
    dc.ulval   = 0;
    dc.cuval   = 0;
    dc.juval   = YES;
    dc.adjval  = ADJ_BOTH;
    dc.boval   = 0;
    dc.bsflg   = FALSE;
```

```
dc.prflg   = TRUE;
dc.sprdir  = 0;
dc.flevel  = 0;
dc.lastfnt = 1;
dc.thisfnt = 1;
dc.escon   = YES;
dc.pgchr   = '%';
dc.cmdchr  = '.';
dc.escchr  = '\\';
dc.nobrchr  = '\'';
for (i = 0; i < 26; ++i)
     dc.nr[i] = 0;
for (i = 0; i < 26; ++i)
     dc.nrauto[i] = 1;
for (i = 0; i < 26; ++i)
     dc.nrfmt[i] = '1';


/*
 *   initialize internal regs. first zero out...
 */
for (i = 0; i < MAXREGS; i++)
{
     rg[i].rname[0] = rg[i].rname[1] = rg[i].rname[2] = rg[i].rname[3] = '\0';
     rg[i].rauto = 1;
     rg[i].rval  = 0;
     rg[i].rflag = RF_READ;
     rg[i].rfmt  = '1';
}



/*
 *   this should be checked...
 */
ctim = ctime (&dum);


/*
 *   predefined regs. these are read/write:
 */
i = 0;

strcpy (rg[i].rname, "%");       /* current page */
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ | RF_WRITE;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, "ct");      /* character type */
rg[i].rauto = 1;
rg[i].rval  = 0;
```

```
rg[i].rflag = RF_READ | RF_WRITE;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, "dl");      /* width of last complete di */
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ | RF_WRITE;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, "dn");      /* height of last complete di */
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ | RF_WRITE;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, "dw");      /* day of week (1-7) */
rg[i].rval  = 0;
if      (!strncmp (&ctim[0], "Sun", 3))
     rg[i].rval  = 1;
else if (!strncmp (&ctim[0], "Mon", 3))
     rg[i].rval  = 2;
else if (!strncmp (&ctim[0], "Tue", 3))
     rg[i].rval  = 3;
else if (!strncmp (&ctim[0], "Wed", 3))
     rg[i].rval  = 4;
else if (!strncmp (&ctim[0], "Thu", 3))
     rg[i].rval  = 5;
else if (!strncmp (&ctim[0], "Fri", 3))
     rg[i].rval  = 6;
else if (!strncmp (&ctim[0], "Sat", 3))
     rg[i].rval  = 7;
rg[i].rauto = 1;
rg[i].rflag = RF_READ | RF_WRITE;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, "dy");      /* day of month (1-31) */
rg[i].rauto = 1;
rg[i].rval  = atoi (&ctim[8]);
rg[i].rflag = RF_READ | RF_WRITE;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, "hp");      /* current h pos on input */
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ | RF_WRITE;
rg[i].rfmt  = '1';
i++;
```

```
strcpy (rg[i].rname, "ln");      /* output line num */
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ | RF_WRITE;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, "mo");      /* current month (1-12) */
rg[i].rval  = 0;
if      (!strncmp (&ctim[4], "Jan", 3))
     rg[i].rval  = 1;
else if (!strncmp (&ctim[4], "Feb", 3))
     rg[i].rval  = 2;
else if (!strncmp (&ctim[4], "Mar", 3))
     rg[i].rval  = 3;
else if (!strncmp (&ctim[4], "Apr", 3))
     rg[i].rval  = 4;
else if (!strncmp (&ctim[4], "May", 3))
     rg[i].rval  = 5;
else if (!strncmp (&ctim[4], "Jun", 3))
     rg[i].rval  = 6;
else if (!strncmp (&ctim[4], "Jul", 3))
     rg[i].rval  = 7;
else if (!strncmp (&ctim[4], "Aug", 3))
     rg[i].rval  = 8;
else if (!strncmp (&ctim[4], "Sep", 3))
     rg[i].rval  = 9;
else if (!strncmp (&ctim[4], "Oct", 3))
     rg[i].rval  = 10;
else if (!strncmp (&ctim[4], "Nov", 3))
     rg[i].rval  = 11;
else if (!strncmp (&ctim[4], "Dec", 3))
     rg[i].rval  = 12;
rg[i].rauto = 1;
rg[i].rflag = RF_READ | RF_WRITE;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, "nl");      /* v pos of last base-line */
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ | RF_WRITE;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, "sb");      /* depth of str below base */
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ | RF_WRITE;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, "st");      /* height of str above base */
```

```
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ | RF_WRITE;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, "yr");      /* last 2 dig of current year*/
rg[i].rauto = 1;
rg[i].rval  = atoi (&ctim[22]);
rg[i].rflag = RF_READ | RF_WRITE;
rg[i].rfmt  = '1';
i++;


/*
 *   these are read only:
 */
strcpy (rg[i].rname, ".$");      /* num of args at current macro*/
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".A");      /* 1 for nroff */
rg[i].rauto = 1;
rg[i].rval  = 1;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".H");      /* hor resolution */
rg[i].rauto = 1;
rg[i].rval  = 1;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".T");      /* 1 for troff */
rg[i].rauto = 0;
rg[i].rval  = 0;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".V");      /* vert resolution */
rg[i].rauto = 1;
rg[i].rval  = 1;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".a");
```

```
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".c");
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".d");
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".f");      /* current font (1-4) */
rg[i].rauto = 1;
rg[i].rval  = 1;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".h");
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".i");      /* current indent */
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".l");      /* current line length */
rg[i].rauto = 1;
rg[i].rval  = PAGEWIDTH - 1;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".n");
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
```

```
i++;

strcpy (rg[i].rname, ".o");      /* current offset */
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".p");      /* current page len */
rg[i].rauto = 1;
rg[i].rval  = PAGELEN;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".s");      /* current point size */
rg[i].rauto = 1;
rg[i].rval  = 1;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".t");
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".u");
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".v");      /* current v line spacing */
rg[i].rauto = 1;
rg[i].rval  = 1;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".w");      /* width of prev char */
rg[i].rauto = 1;
rg[i].rval  = 1;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".x");
rg[i].rauto = 1;
```

```
rg[i].rval  = 0;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".y");
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';
i++;

strcpy (rg[i].rname, ".z");
rg[i].rauto = 1;
rg[i].rval  = 0;
rg[i].rflag = RF_READ;
rg[i].rfmt  = '1';

pg.curpag   = 0;
pg.newpag   = 1;
pg.lineno   = 0;
pg.plval    = PAGELEN;
pg.m1val    = 2;
pg.m2val    = 2;
pg.m3val    = 2;
pg.m4val    = 2;
pg.bottom   = pg.plval - pg.m4val - pg.m3val;
pg.offset   = 0;
pg.frstpg   = 0;
pg.lastpg   = 30000;
pg.ehead[0] = pg.ohead[0] = '\n';
pg.efoot[0] = pg.ofoot[0] = '\n';
for (i = 1; i < MAXLINE; ++i)
{
     pg.ehead[i] = pg.ohead[i] = EOS;
     pg.efoot[i] = pg.ofoot[i] = EOS;
}
pg.ehlim[LEFT]  = pg.ohlim[LEFT]  = dc.inval;
pg.eflim[LEFT]  = pg.oflim[LEFT]  = dc.inval;
pg.ehlim[RIGHT] = pg.ohlim[RIGHT] = dc.rmval;
pg.eflim[RIGHT] = pg.oflim[RIGHT] = dc.rmval;

co.outp   = 0;
co.outw   = 0;
co.outwds = 0;
co.lpr    = FALSE;
for (i = 0; i < MAXLINE; ++i)
     co.outbuf[i] = EOS;

for (i = 0; i < MXMDEF; ++i)
     mac.mnames[i] = NULL;
for (i = 0; i < MACBUF; ++i)
     mac.mb[i] = '\0';
```

```
     for (i = 0; i < MAXLINE; ++i)
          mac.pbb[i] = '\0';
     mac.lastp = 0;
     mac.emb   = &mac.mb[0];
     mac.ppb   = NULL;
}




/*-----------------------------*/
/*   pswitch           */
/*-----------------------------*/
pswitch (p, p2, q)
register char  *p;
register char  *p2;
register int   *q;
{

/*
 *   process switch values from command line
 */

     int     swgood;
     char    mfile[128];
     char   *ptmac;

     swgood = TRUE;
     if (*p == '-')
     {
          /*
           *   since is STILL use the goofy atari/dri xmain code, i
           *   look for both upper and lower case. if you use dLibs
           *   (and if its startup code does not ucase the cmd line),
           *   you can probably look for just lower case. gulam and
           *   other shells typically don't change case of cmd line.
           */
          switch (*++p)
          {
          case 0:                     /* stdin */
               break;

          case 'a':                   /* font changes */
          case 'A':
               dc.dofnt = NO;
               break;

          case 'b':                   /* backspace */
          case 'B':
               dc.bsflg = TRUE;
               break;

          case 'd':                   /* debug mode */
```

```
case 'D':
     dbg_stream = fopen (dbgfile, "w");
     debugging = TRUE;
     break;

case 'h':                    /* hold screen */
case 'H':
     hold_screen = TRUE;
     break;

case 'l':                    /* to lpr (was P) */
case 'L':
     pout = fopen (printer, "w");
     co.lpr = TRUE;
     break;

case 'm':                    /* macro file */
case 'M':
     /*
      *   build macro file name. start with lib
      *
      *   put c:\lib\tmac in environment so we can
      *   read it here. else use default. if you want
      *   file from cwd, "setenv TMACDIR ." from shell.
      *
      *   we want file names like "tmac.an" (for -man)
      */
     if (ptmac = getenv ("TMACDIR"))
     {
          /*
           *   this is the lib path (e.g. "c:\lib\tmac")
           */
          strcpy (mfile, ptmac);

          /*
           *   this is the prefix (i.e. "\tmac.")
           */
          strcat (mfile, TMACPRE);
     }
     else
          /*
           *   use default lib/prefix (i.e.
           *   "c:\lib\tmac\tmac.")
           */
          strcpy (mfile, TMACFULL);

     /*
      *   finally, add extension (e.g. "an")
      */
     strcat (mfile, ++p);

     /*
      *   open file and read it
```

```
      */
     if ((sofile[0] = fopen (mfile, "r")) == NULL)
     {
          fprintf (stderr,
               "***%s: unable to open macro file %s\n",
               myname, mfile);
          err_exit (-1);
     }
     profile ();
     fclose (sofile[0]);
     break;

case 'o':                    /* output error log */
case 'O':
     if (!p2)
     {
          fprintf (stderr,
               "***%s: no error file specified\n",
               myname);
          err_exit (-1);
     }
     if ((err_stream = fopen (p2, "w")) == NULL)
     {
          fprintf (stderr,
               "***%s: unable to open error file %s\n",
               myname, p2);
          err_exit (-1);
     }
     break;

case 'p':                    /* .po, .pn */
case 'P':
     if (*(p+1) == 'o' || *(p+1) == 'O')  /* -po___ */
     {
          p += 2;
          set (&pg.offset, ctod (p), '1', 0, 0, HUGE);
          set_ireg (".o", pg.offset, 0);
     }
     else if (*(p+1) == 'n' || *(p+1) == 'N')/* -pn___ */
     {
          p += 2;
          set (&pg.curpag, ctod (p) - 1, '1', 0, -HUGE, HUGE);
          pg.newpag = pg.curpag + 1;
          set_ireg ("%", pg.newpag, 0);
     }
     else                    /* -p___ */
     {
          set (&pg.offset, ctod (++p), '1', 0, 0, HUGE);
          set_ireg (".o", pg.offset, 0);
     }
     break;

case 'v':                    /* version */
```

```
          case 'V':
               printf ("%s\n", version);
               *q = TRUE;
               break;

          case '0':                    /* last page */
          case '1':
          case '2':
          case '3':
          case '4':
          case '5':
          case '6':
          case '7':
          case '8':
          case '9':
               pg.lastpg = ctod (p);
               break;

          default:                /* illegal */
               swgood = FALSE;
               break;
          }
     }
     else if (*p == '+')                  /* first page */
     {
          pg.frstpg = ctod (++p);
     }
     else                         /* illegal */
     {
          swgood = FALSE;
     }


     if (swgood == FALSE)
     {
          fprintf (stderr, "***%s: illegal switch %s\n", myname, p);
          return (ERR);
     }

     return (OK);
}




/*------------------------------*/
/*   profile            */
/*------------------------------*/
profile ()
{

/*
 *   process input files from command line
```

```
 */

    char    ibuf[MAXLINE];

    /*
     *   handle nesting of includes (.so)
     */
    for (dc.flevel = 0; dc.flevel >= 0; dc.flevel -= 1)
    {
         while (getlin (ibuf, sofile[dc.flevel]) != EOF)
         {
              /*
               *   if line is a command or text
               */
              if (ibuf[0] == dc.cmdchr)
              {
                   comand (ibuf);
              }
              else
              {
                   /*
                    *   this is a text line. first see if
                    *   first char is space. if it is, break
                    *   line.
                    */
                   if (ibuf[0] == ' ')
                        robrk ();
                   text (ibuf);
              }
         }

         /*
          *   close included file
          */
         if (dc.flevel > 0)
              fclose (sofile[dc.flevel]);
    }
    if (pg.lineno > 0)
         space (HUGE);
}




@//E*O*F main.c//
chmod u=rw,g=r,o=r main.c

echo x - command.c
sed 's/^@//' > "command.c" <<'@//E*O*F command.c//'
/*
 *   command.c - command input parser/processor for nroff text processor
 *
```

```
 *   adapted for atariST/TOS by Bill Rosenkranz 11/89
 *   net: rosenkra@hall.cray.com
 *   CIS: 71460,17
 *   GENIE:   W.ROSENKRANZ
 *
 *   original author:
 *
 *   Stephen L. Browning
 *   5723 North Parker Avenue
 *   Indianapolis, Indiana 46220
 *
 *   history:
 *
 *   - Originally written in BDS C;
 *   - Adapted for standard C by W. N. Paul
 *   - Heavily hacked up to conform to "real" nroff by Bill Rosenkranz
 */

#undef NRO_MAIN                      /* extern globals */

#include <stdio.h>
#include "nroff.h"



/*------------------------------*/
/*   comand             */
/*------------------------------*/
comand (p)
register char  *p;
{

/*
 *   main command processor
 */

    register int  i;
    register int  ct;
    register int  val;
    register int      indx;
    int      newval;
    int      spval;
    char     argtyp;
    char     name[MAXLINE];
    char     macexp[MXMLEN];
    int      tmp;
    char       *pfs;
    char     fs[20];


    /*
     *   get command code
     */
```

```
ct = comtyp (p, macexp);

/*
 *   error?
 */
if (ct == UNKNOWN)
{
     fprintf (err_stream,
          "***%s: unrecognized command %s\n", myname, p);
     return;
}

/*
 *   ignore comments
 */
if (ct == COMMENT)
     return;


/*
 *   do escape expansion on command line args
 */
expesc (p, name);


/*
 *   get value of command
 */
val = getval (p, &argtyp);


/*
 *   do the command
 */
switch (ct)
{
/* set (&param, val, type, defval, minval, maxval) */
case FC:
     /*
      *   field delim/pad chars
      *
      *   .fc [delim] [pad]
      */
     fprintf (err_stream, "***%s: .fc not available\n", myname);
     break;
case TR:
     /*
      *   translate
      *
      *   .tr ab...
      */
     fprintf (err_stream, "***%s: .tr not available\n", myname);
     break;
```

```
case AD:
     /*
      *   adjust
      *
      *   .ad [mode]
      */
     p = skipwd (p);
     p = skipbl (p);

     switch (*p)
     {
     case 'l':
          dc.adjval = ADJ_LEFT;
          dc.juval  = YES;
          break;
     case 'r':
          dc.adjval = ADJ_RIGHT;
          dc.juval  = YES;
          break;
     case 'c':
          dc.adjval = ADJ_CENTER;
          dc.juval  = YES;
          break;
     case 'b':
     case 'n':
          dc.adjval = ADJ_BOTH;
          dc.juval  = YES;
          break;
     default:
          break;
     }
     break;
case AF:
     /*
      *   assign format to number reg
      *
      *   .af R {1,a,A,i,I,0...1}
      */
     p = skipwd (p);
     p = skipbl (p);
     if (!isalpha (*p))
     {
          fprintf (err_stream,
               "***%s: invalid or missing number register name\n",
               myname);
     }
     else
     {
          /*
```

```
          *   number register format is 1,a,A,i,I,0...1
          *   default is 1. for 0001 format, store num dig
          *   or'ed with 0x80, up to 8 digits.
          */
         indx = tolower (*p) - 'a';
         p = skipwd (p);
         p = skipbl (p);
         if (*p == '1')
              dc.nrfmt[indx] = '1';
         else if (*p == 'a')
              dc.nrfmt[indx] = 'a';
         else if (*p == 'A')
              dc.nrfmt[indx] = 'A';
         else if (*p == 'i')
              dc.nrfmt[indx] = 'i';
         else if (*p == 'I')
              dc.nrfmt[indx] = 'I';
         else if (*p == '0')
         {
              for (i = 0; isdigit (p[i]); i++)
                   ;
              dc.nrfmt[indx] = (char) (i);
              if (dc.nrfmt[indx] <= 0)
                   dc.nrfmt[indx] = '1';
              else if (dc.nrfmt[indx] > 8)
              {
                   dc.nrfmt[indx]  = 8;
                   dc.nrfmt[indx] |= 0x80;
              }
              else
                   dc.nrfmt[indx] |= 0x80;

         }
         else
              dc.nrfmt[indx] = '1';
    }
    break;
case BO:
    /*
     *   bold face
     *
     *   .bo [N]
     */
    set (&dc.boval, val, argtyp, 1, 0, HUGE);
    dc.cuval = dc.ulval = 0;
    break;
case BP:
    /*
     *   begin page
     *
     *   .bp [+/-N]
     */
    if (pg.lineno > 0)
```

```
          space (HUGE);
     set (&pg.curpag, val, argtyp, pg.curpag + 1, -HUGE, HUGE);
     pg.newpag = pg.curpag;
     set_ireg ("%", pg.newpag, 0);
     break;
case BR:
     /*
      *   break (page)
      *
      *   .br
      */
     robrk ();
     break;
case BS:
     /*
      *   backspc in output
      *
      *   .bs [N]
      */
     set (&dc.bsflg, val, argtyp, 1, 0, 1);
     break;
case C2:
     /*
      *   nobreak char
      *
      *   .c2 [c=']
      */
     if (argtyp == '\r' || argtyp == '\n')
          dc.nobrchr = '\'';
     else
          dc.nobrchr = argtyp;
     break;
case CC:
     /*
      *   command character
      *
      *   .cc [c=.]
      */
     if (argtyp == '\r' || argtyp == '\n')
          dc.cmdchr = '.';
     else
          dc.cmdchr = argtyp;
     break;
case CE:
     /*
      *   center
      *
      *   .ce [N]
      */
     robrk ();
     set (&dc.ceval, val, argtyp, 1, 0, HUGE);
     break;
case CU:
```

```
     /*
      *   continuous underline
      *
      *   .cu [N]
      */
     set (&dc.cuval, val, argtyp, 1, 0, HUGE);
     dc.ulval = dc.boval = 0;
     break;
case DE:
     /*
      *   define macro
      *
      *   .de name [end]
      */
     defmac (p, sofile[dc.flevel]);
     break;
case DS:
     /*
      *   define string
      *
      *   .ds name string
      */
     defstr (p);
     break;
case EC:
     /*
      *   escape char
      *
      *   .ec [c=\]
      */
     if (argtyp == '\r' || argtyp == '\n')
         dc.escchr = '\\';
     else
         dc.escchr = argtyp;
     dc.escon = YES;
     break;
case EF:
     /*
      *   even footer
      *
      *   .ef "a" "b" "c"
      */
     gettl (p, pg.efoot, &pg.eflim[0]);
     break;
case EH:
     /*
      *   even header
      *
      *   .eh "a" "b" "c"
      */
     gettl (p, pg.ehead, &pg.ehlim[0]);
     break;
case EN:
```

```
     /*
      *   end macro def (should not get one here...)
      *
      *   .en or ..
      */
     fprintf (err_stream, "***%s: missing .de command\n", myname);
     break;
case EO:
     /*
      *   escape off
      *
      *   .eo
      */
     dc.escon = NO;
     break;
case FI:
     /*
      *   fill
      *
      *   .fi
      */
     robrk ();
     dc.fill = YES;
     break;
case FL:
     /*
      *   flush NOW
      *
      *   .fl
      */
     fflush (pout);
     break;
case FO:
     /*
      *   footer
      *
      *   .fo "a" "b" "c"
      */
     gettl (p, pg.efoot, &pg.eflim[0]);
     gettl (p, pg.ofoot, &pg.oflim[0]);
     break;
case FT:
     /*
      *   font change
      *
      *   .ft {R,I,B,S,P}
      *
      *   the way it's implemented here, it causes a break
      *   rather than be environmental...
      */
     p = skipwd (p);
     p = skipbl (p);
     if (!isalpha (*p))
```

```
     {
          fprintf (err_stream,
               "***%s: invalid or missing font name\n",
               myname);
     }
     else
     {
          pfs = &fs[0];

          fontchange (*p, pfs);

          robrk ();
          fflush (pout);
          fprintf (pout, "%s", pfs);
          fflush (pout);
     }
     break;
case TL:
case HE:
     /*
      *   header (both are currently identical. .he is -me)
      *
      *   .tl "a" "b" "c"
      *   .he "a" "b" "c"
      */
     gettl (p, pg.ehead, &pg.ehlim[0]);
     gettl (p, pg.ohead, &pg.ohlim[0]);
     break;
case IN:
     /*
      *   indenting
      *
      *   .in [+/-N]
      */
     set (&dc.inval, val, argtyp, 0, 0, dc.rmval - 1);
     set_ireg (".i", dc.inval, 0);
     dc.tival = dc.inval;
     break;
case JU:
     /*
      *   justify
      *
      *   .ju
      */
     dc.juval = YES;
     break;
case LL:
     /*
      *   line length
      *
      *   .ll [+/-N]
      *   .rm [+/-N]
      */
```

```
    set (&dc.rmval, val, argtyp, PAGEWIDTH, dc.tival + 1, HUGE);
    set (&dc.llval, val, argtyp, PAGEWIDTH, dc.tival + 1, HUGE);
    set_ireg (".l", dc.llval, 0);
    break;
case LS:
    /*
     *   line spacing
     *
     *   .ls [+/-N=+1]
     */
    set (&dc.lsval, val, argtyp, 1, 1, HUGE);
    set_ireg (".v", dc.lsval, 0);
    break;
case LT:
    /*
     *   title length
     *
     *   .lt N
     */
    set (&dc.ltval, val, argtyp, PAGEWIDTH, 0, HUGE);
    pg.ehlim[RIGHT] = dc.ltval;
    pg.ohlim[RIGHT] = dc.ltval;
    break;
case M1:
    /*
     *   topmost margin
     *
     *   .m1 N
     */
    set (&pg.m1val, val, argtyp, 2, 0, HUGE);
    break;
case M2:
    /*
     *   second top margin
     *
     *   .m2 N
     */
    set (&pg.m2val, val, argtyp, 2, 0, HUGE);
    break;
case M3:
    /*
     *   1st bottom margin
     *
     *   .m3 N
     */
    set (&pg.m3val, val, argtyp, 2, 0, HUGE);
    pg.bottom = pg.plval - pg.m4val - pg.m3val;
    break;
case M4:
    /*
     *   bottom-most marg
     *
     *   .m4 N
```

```
	 */
	set (&pg.m4val, val, argtyp, 2, 0, HUGE);
	pg.bottom = pg.plval - pg.m4val - pg.m3val;
	break;
case MACRO:
	/*
	 *   macro expansion
	 *
	 *   (internal)
	 */
	maceval (p, macexp);
	break;
case NA:
	/*
	 *   no adjust
	 *
	 *   .na
	 */
	dc.adjval = ADJ_OFF;
	dc.juval  = NO;
	break;
case NE:
	/*
	 *   need n lines
	 *
	 *   .ne N
	 */
	robrk ();
	if ((pg.bottom - pg.lineno + 1) < (val * dc.lsval))
	{
	    space (HUGE);
	}
	break;
case NF:
	/*
	 *   no fill
	 *
	 *   .nf
	 */
	robrk ();
	dc.fill = NO;
	break;
case NJ:
	/*
	 *   no justify
	 *
	 *   .nj
	 */
	dc.juval = NO;
	break;
case NR:
	/*
	 *   set number reg
```

```
	 *
	 *   .nr R +/-N M
	 */
	p = skipwd (p);
	p = skipbl (p);
	if (!isalpha (*p))
	{
	    fprintf (err_stream,
		"***%s: invalid or missing number register name\n",
		myname);
	}
	else
	{
	    /*
	     *   indx is the register, R, and val is the final
	     *   value (default = 0). getval does skipwd,skipbl
	     */
	    indx = tolower (*p) - 'a';
	    val = getval (p, &argtyp);
	    set (&dc.nr[indx], val, argtyp, 0, -INFINITE, INFINITE);

	    /*
	     *   now get autoincrement M, if any (default = 1).
	     *   getval does skipwd,skipbl
	     */
	    p = skipwd (p);
	    p = skipbl (p);
	    val = getval (p, &argtyp);
	    set (&dc.nrauto[indx], val, '1', 1, -INFINITE, INFINITE);
	}
	break;
case OF:
	/*
	 *   odd footer
	 *
	 *   .of "a" "b" "c"
	 */
	gettl (p, pg.ofoot, &pg.oflim[0]);
	break;
case OH:
	/*
	 *   odd header
	 *
	 *   .oh "a" "b" "c"
	 */
	gettl (p, pg.ohead, &pg.ohlim[0]);
	break;
case PC:
	/*
	 *   page number char
	 *
	 *   .pc [c=NULL]
	 */
```

```
     if (argtyp == '\r' || argtyp == '\n')
          dc.pgchr = EOS;
     else
          dc.pgchr = argtyp;
     break;
case PL:
     /*
      *   page length
      *
      *   .pl N
      */
     set (&pg.plval,
          val,
          argtyp,
          PAGELEN,
          pg.m1val + pg.m2val + pg.m3val + pg.m4val + 1,
          HUGE);
     set_ireg (".p", pg.plval, 0);
     pg.bottom = pg.plval - pg.m3val - pg.m4val;
     break;
case PM:
     /*
      *   print macro names and sizes
      *
      *   .pm [t]
      */
     if (argtyp == '\r' || argtyp == '\n')
          printmac (0);
     else if (argtyp == 't')
          printmac (1);
     else if (argtyp == 'T')
          printmac (2);
     else
          printmac (0);
     break;
case PN:
     /*
      *   page number
      *
      *   .pn N
      */
     tmp = pg.curpag;
     set (&pg.curpag, val - 1, argtyp, tmp, -HUGE, HUGE);
     pg.newpag = pg.curpag + 1;
     set_ireg ("%", pg.newpag, 0);
     break;
case PO:
     /*
      *   page offset
      *
      *   .po N
      */
     set (&pg.offset, val, argtyp, 0, 0, HUGE);
```

```
     set_ireg (".o", pg.offset, 0);
     break;
case RR:
     /*
      *   unset number reg
      *
      *   .rr R
      */
     p = skipwd (p);
     p = skipbl (p);
     if (!isalpha (*p))
     {
          fprintf (err_stream,
               "***%s: invalid or missing number register name\n",
               myname);
     }
     else
     {
          indx = tolower (*p) - 'a';
          val = 0;
          set (&dc.nr[indx], val, argtyp, 0, -HUGE, HUGE);
     }
     break;
case SO:
     /*
      *   source file
      *
      *   .so name
      */
     p = skipwd (p);
     p = skipbl (p);
     if (getwrd (p, name) == 0)
          break;
     if (dc.flevel + 1 >= Nfiles)
     {
          fprintf (err_stream,
               "***%s: .so commands nested too deeply\n",
               myname);
          err_exit (-1);
     }
     if ((sofile[dc.flevel + 1] = fopen (name, "r")) == NULL)
     {
          fprintf (err_stream,
               "***%s: unable to open %s\n", myname, name);
          err_exit (-1);
     }
     dc.flevel += 1;
     break;
case SP:
     /*
      *   space
      *
      *   .sp [N=1]
```

```
          */
          set (&spval, val, argtyp, 1, 0, HUGE);
          space (spval);
          break;
     case TI:
          /*
           *   temporary indent
           *
           *   .ti [+/-N]
           */
          robrk ();
          set (&dc.tival, val, argtyp, 0, 0, dc.rmval);
          break;
     case UL:
          /*
           *   underline
           *
           *   .ul [N]
           */
          set (&dc.ulval, val, argtyp, 0, 1, HUGE);
          dc.cuval = dc.boval = 0;
          break;
     }
}




/*-----------------------------*/
/*   comtyp             */
/*-----------------------------*/
comtyp (p, m)
register char  *p;
char       *m;
{

/*
 *   decodes nro command and returns its associated
 *   value.
 */

     register char c1;
     register char c2;
     char       *s;
     char     macnam[MNLEN];

     /*
      *   skip past dot and any whitespace
      */
     p++;
     while (*p && (*p == ' ' || *p == '\t'))
          p++;
```

```
if (*p == '\0')
    return (COMMENT);

/*
 *  First check to see if the command is a macro.
 *  If it is, truncate to two characters and return
 *  expansion in m.  Note that upper and lower case
 *  characters are handled differently.
 */
getwrd (p, macnam);
macnam[2] = EOS;
if ((s = getmac (macnam)) != NULL)
{
    strcpy (m, s);
    return (MACRO);
}
c1 = *p++;
c2 = *p;
if (c1 == '\\' && c2 == '\"')        return (COMMENT);
if (c1 == 'a' && c2 == 'f')     return (AF);
if (c1 == 'a' && c2 == 'd')     return (AD);
if (c1 == 'b' && c2 == 'o')     return (BO);
if (c1 == 'b' && c2 == 'p')     return (BP);
if (c1 == 'b' && c2 == 'r')     return (BR);
if (c1 == 'b' && c2 == 's')     return (BS);
if (c1 == 'c' && c2 == 'c')     return (CC);
if (c1 == 'c' && c2 == 'e')     return (CE);
if (c1 == 'c' && c2 == 'u')     return (CU);
if (c1 == 'd' && c2 == 'e')     return (DE);
if (c1 == 'd' && c2 == 's')     return (DS);
if (c1 == 'e' && c2 == 'f')     return (EF);
if (c1 == 'e' && c2 == 'c')     return (EC);
if (c1 == 'e' && c2 == 'h')     return (EH);
if (c1 == 'e' && c2 == 'n')     return (EN);
if (c1 == '.')                  return (EN);
if (c1 == 'e' && c2 == 'o')     return (EO);
if (c1 == 'f' && c2 == 'i')     return (FI);
if (c1 == 'f' && c2 == 'l')     return (FL);
if (c1 == 'f' && c2 == 'o')     return (FO);
if (c1 == 'f' && c2 == 't')     return (FT);
if (c1 == 'h' && c2 == 'e')     return (HE);
if (c1 == 'i' && c2 == 'n')     return (IN);
if (c1 == 'j' && c2 == 'u')     return (JU);
if (c1 == 'l' && c2 == 'l')     return (LL);
if (c1 == 'l' && c2 == 's')     return (LS);
if (c1 == 'm' && c2 == '1')     return (M1);
if (c1 == 'm' && c2 == '2')     return (M2);
if (c1 == 'm' && c2 == '3')     return (M3);
if (c1 == 'm' && c2 == '4')     return (M4);
if (c1 == 'n' && c2 == 'a')     return (NA);
if (c1 == 'n' && c2 == 'e')     return (NE);
if (c1 == 'n' && c2 == 'f')     return (NF);
if (c1 == 'n' && c2 == 'j')     return (NJ);
```

```
    if (c1 == 'n' && c2 == 'r')     return (NR);
    if (c1 == 'o' && c2 == 'f')     return (OF);
    if (c1 == 'o' && c2 == 'h')     return (OH);
    if (c1 == 'p' && c2 == 'c')     return (PC);
    if (c1 == 'p' && c2 == 'l')     return (PL);
    if (c1 == 'p' && c2 == 'm')     return (PM);
    if (c1 == 'p' && c2 == 'o')     return (PO);
    if (c1 == 'r' && c2 == 'm')     return (RM);
    if (c1 == 'r' && c2 == 'r')     return (RR);
    if (c1 == 's' && c2 == 'o')     return (SO);
    if (c1 == 's' && c2 == 'p')     return (SP);
    if (c1 == 't' && c2 == 'i')     return (TI);
    if (c1 == 't' && c2 == 'l')     return (TL);
    if (c1 == 'u' && c2 == 'l')     return (UL);

    if (c1 == 'p' && c2 == 'n')     return (PN);
    if (c1 == 'r' && c2 == 'r')     return (RR);
    if (c1 == 'c' && c2 == '2')     return (C2);
    if (c1 == 't' && c2 == 'r')     return (TR);
    if (c1 == 'l' && c2 == 't')     return (LT);
    if (c1 == 'f' && c2 == 'c')     return (FC);

    return (UNKNOWN);
}




/*-----------------------------*/
/*   gettl             */
/*-----------------------------*/
gettl (p, q, limit)
register char  *p;
register char  *q;
int      limit[];
{

/*
 *   get header or footer title
 */

    /*
     *   skip forward a word...
     */
    p = skipwd (p);
    p = skipbl (p);

    /*
     *   copy and set limits
     */
    strcpy (q, p);
    limit[LEFT]  = dc.inval;
```

```
    limit[RIGHT] = dc.rmval;
}




/*------------------------------*/
/*   getval             */
/*------------------------------*/
getval (p, p_argt)
register char  *p;
register char  *p_argt;
{

/*
 *   retrieves optional argument following nro command.
 *   returns positive integer value with sign (if any)
 *   saved in character addressed by p_argt.
 */

    p = skipwd (p);
    p = skipbl (p);
    *p_argt = *p;
    if ((*p == '+') || (*p == '-'))
        ++p;
    return (ctod (p));
}




/*------------------------------*/
/*   set           */
/*------------------------------*/
set (param, val, type, defval, minval, maxval)
register int   *param;
register int  val;
register char type;
register int  defval;
register int  minval;
register int  maxval;
{

/*
 *   set parameter and check range. this is for basically all commands
 *   which take interger args
 *
 *   no param (i.e. \r or \n) means reset default
 *   + means param += val (increment)
 *   - means param -= val (decrement)
 *   anything else makes an assignment within the defined numerical limits
 *
```

```
 *   examples:
 *
 *   .nr a 14  set register 'a' to 14
 *   .nr a +1  increment register 'a' by 1
 *   .nr a         reset register 'a' to default value (0)
 */

    switch (type)
    {
    case '\r':
    case '\n':
         *param = defval;
         break;
    case '+':
         *param += val;
         break;
    case '-':
         *param -= val;
         break;
    default:
         *param = val;
         break;
    }
    *param = min (*param, maxval);
    *param = max (*param, minval);
}




/*-----------------------------*/
/*   set_ireg      */
/*-----------------------------*/
set_ireg (name, val, opt)
register char  *name;
register int  val;
register int  opt;              /* 0=internal, 1=user set */
{

/*
 *   set internal register "name" to val. ret 0 if ok, else -1 if reg not
 *   found or 1 if read only
 */

    register int  nreg;

    nreg = findreg (name);
    if (nreg < 0)
         return (-1);

    if ((rg[nreg].rflag & RF_WRITE) || (opt == 0))
    {
```

```
          rg[nreg].rval = val;

          return (0);
     }

     return (1);
}

@//E*O*F command.c//
chmod u=rw,g=r,o=r command.c

echo x - text.c
sed 's/^@//' > "text.c" <<'@//E*O*F text.c//'
/*
 *   text.c - text output processing portion of nroff word processor
 *
 *   adapted for atariST/TOS by Bill Rosenkranz 11/89
 *   net: rosenkra@hall.cray.com
 *   CIS: 71460,17
 *   GENIE:   W.ROSENKRANZ
 *
 *   original author:
 *
 *   Stephen L. Browning
 *   5723 North Parker Avenue
 *   Indianapolis, Indiana 46220
 *
 *   history:
 *
 *   - Originally written in BDS C;
 *   - Adapted for standard C by W. N. Paul
 *   - Heavily hacked up to conform to "real" nroff by Bill Rosenkranz
 */

#undef NRO_MAIN                      /* extern globals */

#include <stdio.h>
#include "nroff.h"


/*------------------------------*/
/*   text          */
/*------------------------------*/
text (p)
register char  *p;
{

/*
 *   main text processing
 */

     register int  i;
     char     wrdbuf[MAXLINE];
```

```
/*
 *   skip over leading blanks if in fill mode. we indent later.
 *   since leadbl does a robrk, do it if in .nf mode
 */
if (dc.fill == YES)
{
     if (*p == ' ' || *p == '\n' || *p == '\r')
          leadbl (p);
}
else
     robrk ();


/*
 *   expand escape sequences
 */
expesc (p, wrdbuf);


/*
 *   test for how to output
 */
if (dc.ulval > 0)
{
     /*
      *   underline (.ul)
      *
      *   Because of the way underlining is handled,
      *   MAXLINE should be declared to be three times
      *   larger than the longest expected input line
      *   for underlining.  Since many of the character
      *   buffers use this parameter, a lot of memory
      *   can be allocated when it may not really be
      *   needed.  A MAXLINE of 180 would allow about
      *   60 characters in the output line to be
      *   underlined (remember that only alphanumerics
      *   get underlined - no spaces or punctuation).
      */
     underl (p, wrdbuf, MAXLINE);
     --dc.ulval;
}
if (dc.cuval > 0)
{
     /*
      *   continuous underline (.cu)
      */
     underl (p, wrdbuf, MAXLINE);
     --dc.cuval;
}
if (dc.boval > 0)
{
```

```
        /*
         *   bold (.bo)
         */
        bold (p, wrdbuf, MAXLINE);
        --dc.boval;
    }
    if (dc.ceval > 0)
    {
        /*
         *   centered (.ce)
         */
        center (p);
        put (p);
        --dc.ceval;
    }
    else if ((*p == '\r' || *p == '\n') && dc.fill == NO)
    {
        /*
         *   all blank line
         */
        put (p);
    }
    else if (dc.fill == NO)
    {
        /*
         *   unfilled (.nf)
         */
        put (p);
    }
    else
    {
        /*
         *   anything else...
         *
         *   init escape char counter for this line...
         */
        co.outesc = 0;


        /*
         *   get a word and put it out. increment ptr to the next
         *   word.
         */
        while ((i = getwrd (p, wrdbuf)) > 0)
        {
            co.outesc += countesc (wrdbuf);

            putwrd (wrdbuf);
            p += i;
        }
    }
}
```

```
/*------------------------------*/
/*   bold          */
/*------------------------------*/
bold (p0, p1, size)
register char  *p0;
register char  *p1;
int      size;
{

/*
 *   insert bold face text (by overstriking)
 */

     register int  i;
     register int  j;

     j = 0;
     for (i = 0; (p0[i] != '\n') && (j < size - 1); ++i)
     {
          if (isalpha (p0[i]) || isdigit (p0[i]))
          {
               p1[j++] = p0[i];
               p1[j++] = '\b';
          }
          p1[j++] = p0[i];
     }
     p1[j++] = '\n';
     p1[j] = EOS;
     while (*p1 != EOS)
          *p0++ = *p1++;
     *p0 = EOS;
}
```

```
/*------------------------------*/
/*   center             */
/*------------------------------*/
center (p)
register char  *p;
{

/*
 *   center a line by setting tival
 */

     dc.tival = max ((dc.rmval + dc.tival - width (p)) >> 1, 0);
```

```
}



/*------------------------------*/
/*   expand             */
/*------------------------------*/
expand (p0, c, s)
register char  *p0;
char     c;
register char  *s;
{

/*
 *   expand title buffer to include character string
 */

     register char  *p;
     register char  *q;
     register char  *r;
     char     tmp[MAXLINE];

     p = p0;
     q = tmp;
     while (*p != EOS)
     {
          if (*p == c)
          {
               r = s;
               while (*r != EOS)
                    *q++ = *r++;
          }
          else
               *q++ = *p;
          ++p;
     }
     *q = EOS;
     strcpy (p0, tmp);       /* copy it back */
}



/*------------------------------*/
/*   justcntr      */
/*------------------------------*/
justcntr (p, q, limit)
register char  *p;
char       *q;
int      limit[];
{
```

```
/*
 *   center title text into print buffer
 */

     register int  len;

     len = width (p);
     q   = &q[(limit[RIGHT] + limit[LEFT] - len) >> 1];
     while (*p != EOS)
          *q++ = *p++;
}




/*------------------------------*/
/*   justleft      */
/*------------------------------*/
justleft (p, q, limit)
register char  *p;
char        *q;
int       limit;
{

/*
 *   left justify title text into print buffer
 */

     q = &q[limit];
     while (*p != EOS)
          *q++ = *p++;
}




/*------------------------------*/
/*   justrite      */
/*------------------------------*/
justrite (p, q, limit)
register char  *p;
char        *q;
int       limit;
{

/*
 *   right justify title text into print buffer
 */

     register int  len;

     len = width (p);
```

```
     q = &q[limit - len];
     while (*p != EOS)
          *q++ = *p++;
}




/*------------------------------*/
/*   leadbl             */
/*------------------------------*/
leadbl (p)
register char  *p;
{

/*
 *   delete leading blanks, set tival
 */

     register int  i;
     register int  j;

     /*
      *   end current line and reset co struct
      */
     robrk ();

     /*
      *   skip spaces
      */
     for (i = 0; p[i] == ' ' || p[i] == '\t'; ++i)
          ;

     /*
      *   if not end of line, reset current temp indent
      */
     if (p[i] != '\n' && p[i] != '\r')
          dc.tival = i;

     /*
      *   shift string
      */
     for (j = 0; p[i] != EOS; ++j)
          p[j] = p[i++];
     p[j] = EOS;
}
```

```
/*------------------------------*/
/*   pfoot              */
/*------------------------------*/
pfoot ()
{

/*
 *   put out page footer
 */

     if (dc.prflg == TRUE)
     {
          skip (pg.m3val);
          if (pg.m4val > 0)
          {
               if ((pg.curpag % 2) == 0)
               {
                    puttl (pg.efoot, pg.eflim, pg.curpag);
               }
               else
               {
                    puttl (pg.ofoot, pg.oflim, pg.curpag);
               }
               skip (pg.m4val - 1);
          }
     }
}




/*------------------------------*/
/*   phead              */
/*------------------------------*/
phead ()
{

/*
 *   put out page header
 */

     pg.curpag = pg.newpag;
     if (pg.curpag >= pg.frstpg && pg.curpag <= pg.lastpg)
     {
          dc.prflg = TRUE;
     }
     else
     {
          dc.prflg = FALSE;
     }
     ++pg.newpag;
     set_ireg ("%", pg.newpag, 0);
```

```
     if (dc.prflg == TRUE)
     {
          if (pg.m1val > 0)
          {
               skip (pg.m1val - 1);
               if ((pg.curpag % 2) == 0)
               {
                    puttl (pg.ehead, pg.ehlim, pg.curpag);
               }
               else
               {
                    puttl (pg.ohead, pg.ohlim, pg.curpag);
               }
          }
          skip (pg.m2val);
     }
     /*
      *   initialize lineno for the next page
      */
     pg.lineno = pg.m1val + pg.m2val + 1;
     set_ireg ("ln", pg.lineno, 0);
}




/*------------------------------*/
/*   puttl              */
/*------------------------------*/
puttl (p, lim, pgno)
register char  *p;
int       lim[];
int       pgno;
{

/*
 *   put out title or footer
 */

     register int  i;
     char     pn[8];
     char     t[MAXLINE];
     char     h[MAXLINE];
     char     delim;

     itoda (pgno, pn, 6);
     for (i = 0; i < MAXLINE; ++i)
          h[i] = ' ';
     delim = *p++;
     p = getfield (p, t, delim);
     expand (t, dc.pgchr, pn);
     justleft (t, h, lim[LEFT]);
     p = getfield (p, t, delim);
```

```
     expand (t, dc.pgchr, pn);
     justcntr (t, h, lim);
     p = getfield (p, t, delim);
     expand (t, dc.pgchr, pn);
     justrite (t, h, lim[RIGHT]);
     for (i = MAXLINE - 4; h[i] == ' '; --i)
          h[i] = EOS;
     h[++i] = '\n';
     h[++i] = '\r';
     h[++i] = EOS;
     if (strlen (h) > 2)
     {
          for (i = 0; i < pg.offset; ++i)
               prchar (' ', pout);
     }
     putlin (h, pout);
}




/*------------------------------*/
/*   putwrd             */
/*------------------------------*/
putwrd (wrdbuf)
register char  *wrdbuf;
{

/*
 *   put word in output buffer
 */

     register char  *p0;
     register char  *p1;
     int       w;
     int       last;
     int       llval;
     int       nextra;



     /*
      *   check if this word puts us over the limit
      */
     w     = width (wrdbuf);
     last  = strlen (wrdbuf) + co.outp;
     llval = dc.rmval - dc.tival;
     if (((co.outp > 0) && ((co.outw + w) > llval))
     ||   (last > MAXLINE))
     {
          /*
           *   last word exceeds limit so prepare to break line, print
```

```
          *   it, and reset outbuf.
          */
         last -= co.outp;
         if (dc.juval == YES)
         {
              nextra = llval - co.outw + 1;

              /*
               *   Check whether last word was end of
               *   sentence and modify counts so that
               *   it is right justified.
               */
              if (co.outbuf[co.outp - 2] == ' ')
              {
                   --co.outp;
                   ++nextra;
              }
              spread (co.outbuf, co.outp - 1, nextra, co.outwds, co.outesc);
              if ((nextra > 0) && (co.outwds > 1))
              {
                   co.outp += (nextra - 1);
              }
/*            if (co.outesc > 0)
              {
                   co.outp += co.outesc;
              }
*/
         }

         /*
          *   break line, output it, and reset all co members. reset
          *   esc count.
          */
         robrk ();

         co.outesc = countesc (wrdbuf);
    }


    /*
     *   copy the current word to the out buffer which may have been
     *   reset
     */
    p0 = wrdbuf;
    p1 = co.outbuf + co.outp;
    while (*p0 != EOS)
         *p1++ = *p0++;

    co.outp              = last;
    co.outbuf[co.outp++] = ' ';
    co.outw             += w + 1;
    co.outwds           += 1;
}
```

```
/*-----------------------------*/
/*   skip          */
/*-----------------------------*/
skip (n)
register int  n;
{

/*
 *   skips the number of lines specified by n.
 */

     register int  i;

     if (dc.prflg == TRUE && n > 0)
     {
          for (i = 0; i < n; ++i)
          {
               prchar ('\n', pout);
          }
          prchar ('\r', pout);
     }
}




/*-----------------------------*/
/*   spread             */
/*-----------------------------*/
spread (p, outp, nextra, outwds, escapes)
register char  *p;
int      outp;
int      nextra;
int      outwds;
int      escapes;
{

/*
 *   spread words to justify right margin
 */

     register int  i;
     register int  j;
     register int  nb;
     register int  ne;
     register int  nholes;
     int      jmin;
```

```
    /*
     *   quick sanity check...
     */
    if ((nextra <= 0) || (outwds <= 1))
        return;


/*fflush (pout); fprintf (err_stream, "in spread: escapes = %d\n", escapes);
fflush (err_stream);*/


    /*
     *   set up for the spread and do it...
     */
    dc.sprdir = ~dc.sprdir;
    ne        = nextra;
    nholes    = outwds - 1;          /* holes between words */
    i         = outp - 1;            /* last non-blank character */
    j         = min (MAXLINE - 3, i + ne);    /* leave room for CR,LF,EOS */
/*
    j        += escapes;
    if (p[i-1] == 27)
        j += 2;
    j = min (j, MAXLINE - 3);
*/
    while (i < j)
    {
        p[j] = p[i];
        if (p[i] == ' ')
        {
            if (dc.sprdir == 0)
                nb = (ne - 1) / nholes + 1;
            else
                nb = ne / nholes;
            ne -= nb;
            --nholes;
            for (; nb > 0; --nb)
            {
                --j;
                p[j] = ' ';
            }
        }
        --i;
        --j;
    }
}




/*-----------------------------*/
```

```
/*   strkovr           */
/*-----------------------------*/
strkovr (p, q)
register char  *p;
register char  *q;
{

/*
 *   split overstrikes (backspaces) into seperate buffer
 */

     register char  *pp;
     int      bsflg;

     bsflg = FALSE;
     pp = p;
     while (*p != EOS)
     {
          *q = ' ';
          *pp = *p;
          ++p;
          if (*p == '\b')
          {
               if (*pp >= ' ' && *pp <= '~')
               {
                    bsflg = TRUE;
                    *q = *pp;
                    ++p;
                    *pp = *p;
                    ++p;
               }
          }
          ++q;
          ++pp;
     }
     *q++ = '\r';
     *q = *pp = EOS;

     return (bsflg);
}




/*-----------------------------*/
/*   underl            */
/*-----------------------------*/
underl (p0, p1, size)
register char  *p0;
register char  *p1;
int      size;
{
```

```
/*
 *   underline a line
 */

     register int  i;
     register int  j;

     j = 0;
     for (i = 0; (p0[i] != '\n') && (j < size - 1); ++i)
     {
          if (p0[i] >= ' ' && p0[i] <= '~')
          {
               if (isalpha (p0[i]) || isdigit (p0[i]) || dc.cuval > 0)
               {
                    p1[j++] = '_';
                    p1[j++] = '\b';
               }
          }
          p1[j++] = p0[i];
     }
     p1[j++] = '\n';
     p1[j] = EOS;
     while (*p1 != EOS)
          *p0++ = *p1++;
     *p0 = EOS;
}




/*-----------------------------*/
/*   width              */
/*-----------------------------*/
width (s)
register char  *s;
{

/*
 *   compute width of character string
 */

     register int  w;

     w = 0;
     while (*s != EOS)
     {
          if (*s == '\b')
               --w;
          else if (*s != '\n' && *s != '\r')
               ++w;
          ++s;
     }
```

```
    return (w);
}
@//E*O*F text.c//
chmod u=rw,g=r,o=r text.c

echo x - io.c
sed 's/^@//' > "io.c" <<'@//E*O*F io.c//'
/*
 *   io.c - low level I/O processing portion of nroff word processor
 *
 *   adapted for atariST/TOS by Bill Rosenkranz 11/89
 *   net: rosenkra@hall.cray.com
 *   CIS: 71460,17
 *   GENIE:   W.ROSENKRANZ
 *
 *   original author:
 *
 *   Stephen L. Browning
 *   5723 North Parker Avenue
 *   Indianapolis, Indiana 46220
 *
 *   history:
 *
 *   - Originally written in BDS C;
 *   - Adapted for standard C by W. N. Paul
 *   - Heavily hacked up to conform to "real" nroff by Bill Rosenkranz
 */

#undef NRO_MAIN                      /* extern globals */

#include <stdio.h>
#include "nroff.h"

/*------------------------------*/
/*   getlin             */
/*------------------------------*/
getlin (p, in_buf)
char   *p;
FILE   *in_buf;
{

/*
 *   retrieve one line of input text
 */

    register char  *q;
    register int  i;
    int      c;
    int      nreg;

    q = p;
    for (i = 0; i < MAXLINE - 1; ++i)
```

```
     {
          c = ngetc (in_buf);
          if (c == EOF)
          {
               *q = EOS;
               c  = strlen (p);
               return (c == 0 ? EOF : c);
          }
          *q++ = c;
          if (c == '\n')
               break;
     }
     *q = EOS;

     nreg = findreg (".c");
     if (nreg > 0)
          set_ireg (".c", rg[nreg].rval + 1, 0);

     return (strlen (p));
}




/*------------------------------*/
/*   ngetc              */
/*------------------------------*/
ngetc (infp)
FILE   *infp;
{

/*
 *   get character from input file or push back buffer
 */

     register int  c;

     if (mac.ppb >= &mac.pbb[0])
          c = *mac.ppb--;
     else
          c = getc (infp);

     return (c);
}



/*------------------------------*/
/*   pbstr              */
/*------------------------------*/
pbstr (p)
```

```
char    p[];
{

/*
 *   Push back string into input stream
 */

     register int  i;

     /*
      *   if string is null, we do nothing
      */
     for (i = strlen (p) - 1; i >= 0; --i)
     {
          putbak (p[i]);
     }
}




/*------------------------------*/
/*   putbak             */
/*------------------------------*/
putbak (c)
char    c;
{

/*
 *   Push character back into input stream. we use the push-back buffer
 *   stored with macros.
 */

     if (mac.ppb < &(mac.pbb[0]))
     {
          mac.ppb = &(mac.pbb[0]);
          *mac.ppb = c;
     }
     else
     {
          if (mac.ppb >= &mac.pbb[MAXLINE - 1])
          {
               fprintf (err_stream,
                    "***%s: push back buffer overflow\n", myname);
               err_exit (-1);
          }
          *++(mac.ppb) = c;
     }
}
```

```
/*------------------------------*/
/*   prchar             */
/*------------------------------*/
prchar (c, fp)
char    c;
FILE   *fp;
{

/*
 *   print character with test for printer
 */

    putc (c, fp);
}
```

```
/*------------------------------*/
/*   put           */
/*------------------------------*/
put (p)
char   *p;
{

/*
 *   put out line with proper spacing and indenting
 */

    register int  j;
    char     os[MAXLINE];

    if (pg.lineno == 0 || pg.lineno > pg.bottom)
    {
        phead ();
    }
    if (dc.prflg == TRUE)
    {
        if (!dc.bsflg)
        {
            if (strkovr (p, os) == TRUE)
            {
                for (j = 0; j < pg.offset; ++j)
                    prchar (' ', pout);
                for (j = 0; j < dc.tival; ++j)
                    prchar (' ', pout);
                putlin (os, pout);
            }
        }
        for (j = 0; j < pg.offset; ++j)
            prchar (' ', pout);
```

```
          for (j = 0; j < dc.tival; ++j)
               prchar (' ', pout);
          putlin (p, pout);
     }
     dc.tival = dc.inval;
     skip (min (dc.lsval - 1, pg.bottom - pg.lineno));
     pg.lineno = pg.lineno + dc.lsval;
     set_ireg ("ln", pg.lineno, 0);
     if (pg.lineno > pg.bottom)
          pfoot ();
}




/*------------------------------*/
/*   putlin             */
/*------------------------------*/
putlin (p, pbuf)
register char  *p;
FILE        *pbuf;
{

/*
 *   output a null terminated string to the file
 *   specified by pbuf.
 */

     while (*p != EOS)
          prchar (*p++, pbuf);
}







@//E*O*F io.c//
chmod u=rw,g=r,o=r io.c

exit 0


From 9118634228181567603
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,28844e3628b88c73,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-11-14 20:42:07 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!wuarchive!texbell!texsun!
newstop!sun!sunfedcomm!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
```

Subject: v03i010: nroff -- Partial clone of UNIX "nroff" formatter part03/03
Keywords: shar, C
Message-ID: <34766@grapevine.uucp>
Date: 15 Nov 89 04:42:07 GMT
Sender: daemon@sunfedcomm
Reply-To: rosenkra@hall.cray.com (Bill Rosenkranz)
Lines: 2672
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: rosenkra@hall.cray.com (Bill Rosenkranz)
Posting-number: Volume 3, Issue 10
Archive-name: nroff/part03

part03/03

-bill
rosenkra@hall.cray.com
rosenkra@boston.cray.com

```
-------------cut here 0----------------------cut here-----------------------
#!/bin/sh
# This is a shell archive.  Remove anything before this line,
# then unpack it by saving it in a file and typing "sh file".
#
# Wrapped by hall!rosenkra on Sun Nov 12 16:35:37 CST 1989
# Contents:  escape.c low.c macros.c strings.c

echo x - escape.c
sed 's/^@//' > "escape.c" <<'@//E*O*F escape.c//'
/*
 *   escape.c - Escape and special character input processing portion of
 *              nroff word processor
 *
 *   adapted for atariST/TOS by Bill Rosenkranz 11/89
 *   net: rosenkra@hall.cray.com
 *   CIS: 71460,17
 *   GENIE:   W.ROSENKRANZ
 *
 *   original author:
 *
 *   Stephen L. Browning
 *   5723 North Parker Avenue
 *   Indianapolis, Indiana 46220
 *
 *   history:
 *
 *   - Originally written in BDS C;
 *   - Adapted for standard C by W. N. Paul
 *   - Heavily hacked up to conform to "real" nroff by Bill Rosenkranz
 */

#undef NRO_MAIN                     /* extern globals */
```

```
#include <stdio.h>
#include "nroff.h"


/*-----------------------------*/
/*   expesc            */
/*-----------------------------*/
expesc (p, q)
char   *p;
char   *q;
{

/*
 *   Expand escape sequences
 */

     register char  *s;
     register char  *t;
     register char  *pstr;
     register int  i;
     register int  val;
     register int  autoinc;
     char     c;
     char     fs[5];                  /* for font change */
     char     nrstr[20];
     char     fmt[20];
     char     name[10];
     int      nreg;
     char       *pfs;
     int      inc;
     int      tmp;
     char     delim;


     s = p;
     t = q;


     /*
      *   if escape parsing is not on, just copy string
      */
     if (dc.escon == NO)
     {
          while (*s != EOS)
          {
               *t++ = *s++;
          }
          *t = EOS;
          strcpy (p, q);

          return;
     }
```

```
/*
 *   do it...
 */
while (*s != EOS)
{
     if (*s != dc.escchr)
     {
          /*
           *   not esc, continue...
           */
          *t++ = *s++;
     }


     else if (*(s + 1) == dc.escchr)
     {
          /*
           *   \\            escape escape
           */
          *t++ = *s++;
          ++s;
     }


     else if (*(s + 1) == 'n')
     {
          /*
           *   \nx, \n(xx         register
           *
           *   first check for \n+... or \n-... (either form)
           */
          s += 2;
          autoinc = 0;
          if (*s == '+')
          {
               autoinc = 1;
               s += 1;
          }
          if (*s == '-')
          {
               autoinc = -1;
               s += 1;
          }



          /*
           *   was this \nx or \n(xx form?
           */
          if (isalpha (*s))
          {
```

```
/*
 *   \nx form. find reg (a-z)
 */
nreg = tolower (*s) - 'a';


/*
 *   was this \n+x or \n-x? if so, do the
 *   auto incr
 */
if (autoinc > 0)
     dc.nr[nreg] += dc.nrauto[nreg];
else if (autoinc < 0)
     dc.nr[nreg] -= dc.nrauto[nreg];

/*
 *   display format
 */
if (dc.nrfmt[nreg] == '1')
{
     /*
      *   normal decimal digits
      */
     t += itoda (dc.nr[nreg], t, 6) - 1;
}
else if (dc.nrfmt[nreg] == 'i')
{
     /*
      *   lower roman
      */
     t += itoroman (dc.nr[nreg], t, 24) - 1;
}
else if (dc.nrfmt[nreg] == 'I')
{
     /*
      *   upper roman
      */
     t += itoROMAN (dc.nr[nreg], t, 24) - 1;
}
else if (dc.nrfmt[nreg] == 'a')
{
     /*
      *   lower letters
      */
     t += itoletter (dc.nr[nreg], t, 12) - 1;
}
else if (dc.nrfmt[nreg] == 'A')
{
     /*
      *   upper letters
      */
     t += itoLETTER (dc.nr[nreg], t, 12) - 1;
}
```

```
     else if (dc.nrfmt[nreg] & 0x80)
     {
          /*
           *   zero-filled decimal
           */
          sprintf (fmt, "%%0%dld",
               (int)(dc.nrfmt[nreg] & 0x7F));
          fmt[5] = '\0';
          sprintf (nrstr, fmt, (long) dc.nr[nreg]);
          tmp = dc.nrfmt[nreg] & 0x7F;
          nrstr[tmp] = '\0';

          strcpy (t, nrstr);
          t += strlen (nrstr);
     }
     else
     {
          /*
           *   normal (default)
           */
          t += itoda (dc.nr[nreg], t, 6) - 1;
     }
     ++s;
}
else if (*s == '%')
{
     /*
      *   \n% form. find index into reg struct
      */
     nreg = findreg ("%");
     if (nreg < 0)
     {
          fprintf (err_stream,
               "***%s: no register match\n",
               myname);
          err_exit (-1);
     }


     /*
      *   was this \n+% or \n-%? if so, do the
      *   auto incr
      */
     if (autoinc > 0)
          rg[nreg].rval += rg[nreg].rauto;
     else if (autoinc < 0)
          rg[nreg].rval -= rg[nreg].rauto;


     /*
      *   display format
      */
     if (rg[nreg].rfmt == '1')
```

```
{
     /*
      *   normal decimal digits
      */
     t += itoda (rg[nreg].rval, t, 6) - 1;
}
else if (rg[nreg].rfmt == 'i')
{
     /*
      *   lower roman
      */
     t += itoroman (rg[nreg].rval, t, 24) - 1;
}
else if (rg[nreg].rfmt == 'I')
{
     /*
      *   upper roman
      */
     t += itoROMAN (rg[nreg].rval, t, 24) - 1;
}
else if (rg[nreg].rfmt == 'a')
{
     /*
      *   lower letters
      */
     t += itoletter (rg[nreg].rval, t, 12) - 1;
}
else if (rg[nreg].rfmt == 'A')
{
     /*
      *   upper letters
      */
     t += itoLETTER (rg[nreg].rval, t, 12) - 1;
}
else if (rg[nreg].rfmt & 0x80)
{
     /*
      *   zero-filled decimal
      */
     sprintf (fmt, "%%0%dld",
          (int)(rg[nreg].rfmt & 0x7F));
     fmt[5] = '\0';
     sprintf (nrstr, fmt, (long) rg[nreg].rval);
     tmp = rg[nreg].rfmt & 0x7F;
     nrstr[tmp] = '\0';

     strcpy (t, nrstr);
     t += strlen (nrstr);
}
else
{
     /*
      *   normal (default)
```

```
          */
          t += itoda (rg[nreg].rval, t, 6) - 1;
     }
     s += 1;
}
else if (*s == '(')
{
     /*
      *   \n(xx form. find index into reg struct
      */
     s += 1;
     name[0] = *s;
     name[1] = *(s + 1);
     if (name[1] == ' '  || name[1] == '\t'
     ||  name[1] == '\n' || name[1] == '\r')
          name[1] = '\0';
     name[2] = '\0';
     nreg = findreg (name);
     if (nreg < 0)
     {
          fprintf (err_stream,
               "***%s: no register match\n",
               myname);
          err_exit (-1);
     }


     /*
      *   was this \n+(xx or \n-(xx? if so, do the
      *   auto incr
      */
     if (rg[nreg].rflag & RF_WRITE)
     {
          if (autoinc > 0)
               rg[nreg].rval += rg[nreg].rauto;
          else if (autoinc < 0)
               rg[nreg].rval -= rg[nreg].rauto;
     }


     /*
      *   display format
      */
     if (rg[nreg].rfmt == '1')
     {
          /*
           *   normal decimal digits
           */
          t += itoda (rg[nreg].rval, t, 6) - 1;
     }
     else if (rg[nreg].rfmt == 'i')
     {
          /*
```

```
                 *   lower roman
                 */
                t += itoroman (rg[nreg].rval, t, 24) - 1;
            }
            else if (rg[nreg].rfmt == 'I')
            {
                /*
                 *   upper roman
                 */
                t += itoROMAN (rg[nreg].rval, t, 24) - 1;
            }
            else if (rg[nreg].rfmt == 'a')
            {
                /*
                 *   lower letters
                 */
                t += itoletter (rg[nreg].rval, t, 12) - 1;
            }
            else if (rg[nreg].rfmt == 'A')
            {
                /*
                 *   upper letters
                 */
                t += itoLETTER (rg[nreg].rval, t, 12) - 1;
            }
            else if (rg[nreg].rfmt & 0x80)
            {
                /*
                 *   zero-filled decimal
                 */
                sprintf (fmt, "%%0%dld",
                     (int)(rg[nreg].rfmt & 0x7F));
                fmt[5] = '\0';
                sprintf (nrstr, fmt, (long) rg[nreg].rval);
                tmp = rg[nreg].rfmt & 0x7F;
                nrstr[tmp] = '\0';

                strcpy (t, nrstr);
                t += strlen (nrstr);
            }
            else
            {
                /*
                 *   normal (default)
                 */
                t += itoda (rg[nreg].rval, t, 6) - 1;
            }
            s += 2;
        }
    }


    else if (*(s + 1) == '\"')
```

```
{
     /*
      *   \"            comment
      */
     *s = EOS;
     *t = *s;

     return;
}


else if (*(s + 1) == '*')
{
     /*
      *   \*x, \*(xx        string
      */
     s += 2;
     if (*s == '(')
     {
          /*
           *   \*(xx form
           */
          s += 1;
          name[0] = *s;
          name[1] = *(s + 1);
          name[2] = '\0';
          pstr = getstr (name);
          if (!pstr)
          {
               fprintf (err_stream,
                    "***%s: string not found\n",
                    myname);
               err_exit (-1);
          }
          while (*pstr)
               *t++ = *pstr++;
          s += 2;
     }
     else
     {
          /*
           *   \*x form
           */
          name[0] = *s;
          name[1] = '\0';
          pstr = getstr (name);
          if (!pstr)
          {
               fprintf (err_stream,
                    "***%s: string not found\n",
                    myname);
               err_exit (-1);
          }
```

```
          while (*pstr)
               *t++ = *pstr++;
          s += 1;
     }
}


else if (*(s + 1) == 'f')
{
     /*
      *   \fx      font
      */
     s += 2;
     pfs = fs;     /* set up ret string */
     fs[0] = '\0';

     /*
      *  it parses 1-2 char of s and returns esc seq for
      *  \fB and \fR (\fI is same as \fB)
      */
     fontchange (*s, pfs);

     /*
      *   imbed the atari (vt52) escape seq
      */
     while (*pfs)
          *t++ = *pfs++;
     ++s;          /* skip B,I,R,S,P */
}


else if (*(s + 1) == '(')
{
     /*
      *   \(xx     special char
      */
     s  += 2;

     /*
      *   it returns num char to skip and sets c to
      *   the ascii value of the char
      */
     inc = specialchar (s, &c);

     /*
      *   skip proper num char in s and add c to target
      */
     if (inc)
     {
          s   += inc;
          *t++ = c;
     }
}
```

```
else if (*(s + 1) == 'e')
{
     /*
      *   \e       printable version of escape
      */
     *t++ = dc.escchr;
     s   += 2;
}


else if (*(s + 1) == '`')
{
     /*
      *   \`       grave, like \(ga
      */
     *t++ = 0x60;
     s  += 2;
}


else if (*(s + 1) == '\'')
{
     /*
      *   \'       accute, like \(aa
      */
     s  += 2;
     *t++ = 0xBA;
}


else if (*(s + 1) == '-')
{
     /*
      *   \-       minus
      */
     s  += 2;
     *t++ = 0x2D;
}


else if (*(s + 1) == '.')
{
     /*
      *   \.       period
      */
     s  += 2;
     *t++ = 0x2E;
}


else if (*(s + 1) == ' ')
```

```
{
     /*
      *   \(space) space
      */
     s  += 2;
     *t++ = 0x20;
}


else if (*(s + 1) == '0')
{
     /*
      *   \0       digital width space
      */
     s  += 2;
     *t++ = 0x20;
}


else if (*(s + 1) == '|')
{
     /*
      *   \|       narrow width char (0 in nroff)
      */
     s  += 2;
}


else if (*(s + 1) == '^')
{
     /*
      *   \^       narrow width char (0 in nroff)
      */
     s  += 2;
}


else if (*(s + 1) == '&')
{
     /*
      *   \&       non-printing zero width
      */
     s  += 2;
}


else if (*(s + 1) == '!')
{
     /*
      *   \!       transparent copy line
      */
     s  += 2;
}
```

```
else if (*(s + 1) == '$')
{
     /*
      *   \$N interpolate arg 1<=N<=9
      */
     s  += 2;
}


else if (*(s + 1) == '%')
{
     /*
      *   \%       hyphen
      */
     s  += 2;
     *t++ = 0x2D;
     *t++ = 0x2D;
}


else if (*(s + 1) == 'a')
{
     /*
      *   \a
      */
     s  += 2;
}


else if (*(s + 1) == 'b')
{
     /*
      *   \b'abc...'
      */
     s  += 2;
}


else if (*(s + 1) == 'c')
{
     /*
      *   \c
      */
     s  += 2;
}


else if (*(s + 1) == 'd')
{
     /*
      *   \d
```

```
     */
     s  += 2;
}


else if (*(s + 1) == 'h')
{
     /*
      *   \h'N'    horiz motion
      */
     s    += 2;
     delim = *s++;
     val   = atoi (s);
     for (i = 0; i < val; i++)
          *t++ = ' ';
     while (*s != delim)
     {
          if (*s == 0)
               break;
          s++;
     }
     if (*s)
          s++;

}


else if (*(s + 1) == 'k')
{
     /*
      *   \kx
      */
     s  += 2;
}


else if (*(s + 1) == 'l')
{
     /*
      *   \l'Nc'
      */
     s  += 2;
}


else if (*(s + 1) == 'L')
{
     /*
      *   \L'Nc'
      */
     s  += 2;
}
```

```
else if (*(s + 1) == 'o')
{
     /*
      *   \o'abc...'        overstrike
      */
     s  += 2;
     delim = *s++;
     while (*s != EOS && *s != delim)
     {
          *t++ = *s++;
          *t++ = 0x08;
     }
     s++;
}


else if (*(s + 1) == 'p')
{
     /*
      *   \p
      */
     s  += 2;
}


else if (*(s + 1) == 'r')
{
     /*
      *   \r
      */
     s  += 2;
}


else if (*(s + 1) == 's')
{
     /*
      *   \sN,\s+-N
      */
     s  += 2;
}


else if (*(s + 1) == 't')
{
     /*
      *   \t       horizontal tab
      */
     s  += 2;
     *t++ = 0x09;
}
```

```
else if (*(s + 1) == 'u')
{
     /*
      *   \u
      */
     s  += 2;
}


else if (*(s + 1) == 'v')
{
     /*
      *   \v'N'    vert tab
      */
     s    += 2;
     delim = *s++;
     val   = atoi (s);
     for (i = 0; i < val; i++)
          *t++ = 0x0A;
     while (*s != delim)
     {
          if (*s == 0)
               break;
          s++;
     }
     if (*s)
          s++;
}


else if (*(s + 1) == 'w')
{
     /*
      *   \w'str'
      */
     s  += 2;
}


else if (*(s + 1) == 'x')
{
     /*
      *   \x'N'
      */
     s  += 2;
}


else if (*(s + 1) == 'z')
{
     /*
      *   \zc  print c w/o spacing
```

```
             */
            s  += 2;
            *t++ = *s++;
            *t++ = 0x08;
        }


        else if (*(s + 1) == '{')
        {
            /*
             *   \{
             */
            s  += 2;
        }


        else if (*(s + 1) == '}')
        {
            /*
             *   \}
             */
            s  += 2;
        }


        else if (*(s + 1) == '\n' || *(s + 1) == '\r')
        {
            /*
             *   \(newline)    ignore newline
             */
            s  += 2;
        }


        else
        {
            /*
             *   \X       any other character not above
             */
            s   += 1;
            *t++ = *s++;
        }

    }

    /*
     *   end the string and return it in original buf
     */
    *t = EOS;
    strcpy (p, q);
}
```

```
/*------------------------------*/
/*   specialchar        */
/*------------------------------*/
specialchar (s, c)
register char  *s;
register char  *c;
{

/*
 *   handles \(xx escape sequences for special characters (atari-specific)
 */

    register char c1;
    register char c2;

    c1 = *s;
    c2 = *(s+1);

    /*
     *   symbols
     */
    if (c1 == 'c' && c2 == 'o') {*c = 0xBD; return (2);}
    /* copyrite */
    if (c1 == 'r' && c2 == 'g') {*c = 0xBE; return (2);}
    /* registered */
    if (c1 == 't' && c2 == 'm') {*c = 0xBF; return (2);}
    /* trademark */
    if (c1 == '1' && c2 == '2') {*c = 0xAB; return (2);}
    /* 1/2 */
    if (c1 == '1' && c2 == '4') {*c = 0xAC; return (2);}
    /* 1/4 */
    if (c1 == 'p' && c2 == '2') {*c = 0xFD; return (2);}
    /* ^2 */
    if (c1 == 'p' && c2 == '3') {*c = 0xFE; return (2);}
    /* ^3 */
    if (c1 == 'p' && c2 == 'n') {*c = 0xFC; return (2);}
    /* ^n */
    if (c1 == 'a' && c2 == 'a') {*c = 0xBA; return (2);}
    /* acute */
    if (c1 == 'g' && c2 == 'a') {*c = 0x60; return (2);}
    /* grave */
    if (c1 == 'd' && c2 == 'e') {*c = 0xF8; return (2);}
    /* degree */
    if (c1 == 'd' && c2 == 'g') {*c = 0xBB; return (2);}
    /* dagger */
    if (c1 == 'c' && c2 == 't') {*c = 0x9B; return (2);}
    /* cent */
    if (c1 == 'b' && c2 == 'u') {*c = 0xF9; return (2);}
    /* bullet */
    if (c1 == 'd' && c2 == 't') {*c = 0xFA; return (2);}
    /* dot */
```

```
if (c1 == 'p' && c2 == 'p') {*c = 0xBC; return (2);}
/* paragraph */
if (c1 == '^' && c2 == 'g') {*c = 0x07; return (2);}
/* ring bell */
if (c1 == 'u' && c2 == 'a') {*c = 0x01; return (2);}
/* up arrow */
if (c1 == 'd' && c2 == 'a') {*c = 0x02; return (2);}
/* dn arrow */
if (c1 == '-' && c2 == '>') {*c = 0x03; return (2);}
/* rt arrow */
if (c1 == '<' && c2 == '-') {*c = 0x04; return (2);}
/* lf arrow */
if (c1 == 'd' && c2 == 'i') {*c = 0xF6; return (2);}
/* divide */
if (c1 == 's' && c2 == 'r') {*c = 0xFB; return (2);}
/* sq root */
if (c1 == '=' && c2 == '=') {*c = 0xF0; return (2);}
/* == */
if (c1 == '>' && c2 == '=') {*c = 0xF2; return (2);}
/* >= */
if (c1 == '<' && c2 == '=') {*c = 0xF3; return (2);}
/* <= */
if (c1 == '+' && c2 == '-') {*c = 0xF1; return (2);}
/* +- */
if (c1 == '~' && c2 == '=') {*c = 0xF7; return (2);}
/* ~= */
if (c1 == 'a' && c2 == 'p') {*c = 0x7E; return (2);}
/* approx */
if (c1 == 'n' && c2 == 'o') {*c = 0xAA; return (2);}
/* not */
if (c1 == 'm' && c2 == 'o') {*c = 0xEE; return (2);}
/* member */
if (c1 == 'c' && c2 == 'a') {*c = 0xEF; return (2);}
/* intersect */
if (c1 == 'c' && c2 == 'u') {*c = 0x55; return (2);}
/* union */
if (c1 == 'i' && c2 == '1') {*c = 0xF4; return (2);}
/* integral1 */
if (c1 == 'i' && c2 == '2') {*c = 0xF5; return (2);}
/* integral2 */

/*
 *   greek
 */
if (c1 == '*' && c2 == 'a') {*c = 0xE0; return (2);}
/* alpha */
if (c1 == '*' && c2 == 'b') {*c = 0xE1; return (2);}
/* beta */
if (c1 == '*' && c2 == 'g') {*c = 0xE2; return (2);}
/* gamma */
if (c1 == '*' && c2 == 'd') {*c = 0x7F; return (2);}
/* delta */
if (c1 == '*' && c2 == 's') {*c = 0xE4; return (2);}
```

```
    /* sigma */
    if (c1 == '*' && c2 == 'p') {*c = 0xE3; return (2);}
    /* pi */
    if (c1 == '*' && c2 == 'm') {*c = 0xE6; return (2);}
    /* mu */

    *c = ' ';
    return (0);
}




/*------------------------------*/
/*   fontchange         */
/*------------------------------*/
fontchange (fnt, s)
char fnt;
char   *s;
{

/*
 *   handles \fx font change escapes for R,B,I,S,P (atari-specific)
 *   resets current and last font in dc struct (last used for .ft
 *   with no args)
 */

    int  tmp;

    *s = '\0';
    switch (fnt)
    {
    case 'R':               /* Times Roman */
        if (dc.dofnt == YES)
            strcpy (s, "\33q");
        dc.lastfnt = dc.thisfnt;
        dc.thisfnt = 1;
        break;
    case 'I':               /* Times italic */
        if (dc.dofnt == YES)
            strcpy (s, "\33p");
        dc.lastfnt = dc.thisfnt;
        dc.thisfnt = 2;
        break;
    case 'B':               /* Times bold */
        if (dc.dofnt == YES)
            strcpy (s, "\33p");
        dc.lastfnt = dc.thisfnt;
        dc.thisfnt = 3;
        break;
    case 'S':               /* math/special */
        *s = '\0';
        dc.lastfnt = dc.thisfnt;
```

```
          dc.thisfnt = 4;
          break;
     case 'P':              /* previous (exchange) */
          if (dc.dofnt == YES)
          {
               if (dc.lastfnt == 1)
                    strcpy (s, "\33q");    /* to R */
               else if (dc.lastfnt == 2)
                    strcpy (s, "\33p");    /* to I */
               else if (dc.lastfnt == 3)
                    strcpy (s, "\33p");    /* to B */
               else
                    *s = '\0';         /* nothing */
          }

          tmp        = dc.thisfnt;         /* swap this/last */
          dc.thisfnt = dc.lastfnt;
          dc.lastfnt = tmp;
          break;
     default:
          *s = '\0';
          break;
     }

     set_ireg (".f", dc.thisfnt, 0);
}




/*-----------------------------*/
/*   findreg            */
/*-----------------------------*/
findreg (name)
register char  *name;
{

/*
 *   find register named 'name' in pool. return index into array or -1
 *   if not found.
 */

     register int  i;
     register char  *prname;

     for (i = 0; i < MAXREGS; i++)
     {
          prname = rg[i].rname;
          if (*prname == *name && *(prname + 1) == *(name + 1))
               break;
     }
```

```
    return ((i < MAXREGS) ? i : -1);
}



@//E*O*F escape.c//
chmod u=rw,g=r,o=r escape.c

echo x - low.c
sed 's/^@//' > "low.c" <<'@//E*O*F low.c//'
/*
 *   low.c - misc low-level functions for nroff word processor
 *
 *   adapted for atariST/TOS by Bill Rosenkranz 11/89
 *   net: rosenkra@hall.cray.com
 *   CIS: 71460,17
 *   GENIE:   W.ROSENKRANZ
 *
 *   original author:
 *
 *   Stephen L. Browning
 *   5723 North Parker Avenue
 *   Indianapolis, Indiana 46220
 *
 *   history:
 *
 *   - Originally written in BDS C;
 *   - Adapted for standard C by W. N. Paul
 *   - Heavily hacked up to conform to "real" nroff by Bill Rosenkranz
 */

#undef NRO_MAIN                     /* extern globals */

#include <stdio.h>
#include "nroff.h"



/*------------------------------*/
/*   atod          */
/*------------------------------*/
atod (c)
char    c;
{

/*
 *   convert ascii character to decimal.
 */

    return (((c < '0') || (c > '9')) ? -1 : c - '0');
}
```

```
/*-----------------------------*/
/*   robrk             */
/*-----------------------------*/
robrk ()
{

/*
 *   end current filled line
 */

     if (co.outp > 0)
     {
          co.outbuf[co.outp]   = '\r';
          co.outbuf[co.outp+1] = '\n';
          co.outbuf[co.outp+2] = EOS;
          put (co.outbuf);
     }
     co.outp   = 0;
     co.outw   = 0;
     co.outwds = 0;
     co.outesc = 0;
}




/*-----------------------------*/
/*   ctod          */
/*-----------------------------*/
ctod (p)
register char  *p;
{

/*
 *   convert string to decimal.
 *   processes only positive values.
 */

     register int  val;
     register int  d;

     val = 0;
     while (*p != EOS)
     {
          d = atod (*p++);
          if (d == -1)
               return (val);
          val = 10 * val + d;
     }
```

```
    return (val);
}




/*------------------------------*/
/*   skipbl             */
/*------------------------------*/
char   *skipbl (p)
register char  *p;
{

/*
 *   skip blanks and tabs in character buffer. return ptr to first
 *   non-space or non-tab char. this could mean EOS or \r or \n.
 *   also increments the arg ptr (side effect).
 */

    while (*p == ' ' || *p == '\t')
        ++p;
    return (p);
}




/*------------------------------*/
/*   skipwd             */
/*------------------------------*/
char   *skipwd (p)
register char  *p;
{

/*
 *   skip over word and punctuation. anything but space,\t,\r,\n, and EOS
 *   is skipped. return ptr to the first of these found. also increments
 *   the arg ptr (side effect).
 */

    while (*p != ' ' && *p != '\t' && *p != '\r' && *p != '\n' && *p != EOS)
        ++p;
    return (p);
}




/*------------------------------*/
/*   space              */
/*------------------------------*/
space (n)
```

```
int     n;
{

/*
 *   space vertically n lines. this does header and footer also.
 */

     robrk ();
     if (pg.lineno > pg.bottom)
          return;
     if (pg.lineno == 0)
          phead ();
     skip (min (n, pg.bottom + 1 - pg.lineno));
     pg.lineno += n;
     set_ireg ("ln", pg.lineno, 0);
     if (pg.lineno > pg.bottom)
          pfoot ();
}




/*------------------------------*/
/*   getfield      */
/*------------------------------*/
char   *getfield (p, q, delim)
register char  *p;
register char  *q;
char     delim;
{

/*
 *   get field from title
 */

     while (*p != delim && *p != '\r' && *p != '\n' && *p != EOS)
     {
          *q++ = *p++;
     }
     *q = EOS;
     if (*p == delim)
          ++p;
     return (p);
}




/*------------------------------*/
/*   getwrd             */
/*------------------------------*/
getwrd (p0, p1)
```

```
register char  *p0;
register char  *p1;
{

/*
 *   get non-blank word from p0 into p1.
 *   return number of characters processed.
 */

    register int  i;
    register char  *p;
    char     c;

    /*
     *   init counter...
     */
    i = 0;


    /*
     *   skip leading whitespace
     */
    while (*p0 && (*p0 == ' ' || *p0 == '\t'))
    {
         ++i;
         ++p0;
    }


    /*
     *   set ptr and start to look for end of word
     */
    p = p0;
    while (*p0 != ' ' && *p0 != EOS && *p0 != '\t')
    {
         if (*p0 == '\n' || *p0 == '\r')
              break;
         *p1 = *p0++;
         ++p1;
         ++i;
    }

    c = *(p1 - 1);
    if (c == '"')
         c = *(p1 - 2);
    if (c == '?' || c == '!')
    {
         *p1++ = ' ';
         ++i;
    }
    if (c == '.' && (*p0 == '\n' || *p0 == '\r' || islower (*p)))
    {
         *p1++ = ' ';
```

```
          ++i;
     }
     *p1 = EOS;

     return (i);
}




/*------------------------------*/
/*   countesc      */
/*------------------------------*/

#define ESC             27

countesc (p)
register char  *p;
{

/*
 *   count atari escape sequence characters in given null-terminated
 *   string
 */

     register char  *pp;
     register int  num;

     pp  = p;
     num = 0;

     while (*pp != EOS)
     {
          if (*pp == ESC)
          {
               /*
                *   count escape char (atari-specific, vt52)
                *   generally only p,q,b,and c will show up...
                */
               switch (*(pp+1))
               {
               case 'A':          /* ESC-a */
               case 'B':
               case 'C':
               case 'D':
               case 'E':
               case 'H':
               case 'I':
               case 'J':
               case 'K':
               case 'L':
               case 'M':
               case 'd':
```

```
                case 'e':
                case 'f':
                case 'j':
                case 'k':
                case 'l':
                case 'o':
                case 'p':
                case 'q':
                case 'v':
                case 'w':
                     num += 2;
                     break;
                case 'b':          /* ESC-a-b */
                case 'c':
                     num += 3;
                     break;
                case 'Y':          /* ESC-a-b-c */
                     num += 4;
                     break;
                default:
                     num += 1;
                     break;
                }
          }
          pp++;
     }

     return (num);
}



/*------------------------------*/
/*   itoda              */
/*------------------------------*/
itoda (value, p, size)
int      value;
register char  *p;
register int  size;
{

/*
 *   convert integer to decimal ascii string
 */

     register int  i;
     register int    j;
     register int  k;
     register int  aval;
     char     c[20];

     aval = abs (value);
```

```
     c[0] = EOS;
     i = 1;
     do
     {
          c[i++] = (aval % 10) + '0';
          aval /= 10;

     } while (aval > 0 && i <= size);

     if (value < 0 && i <= size)
          c[i++] = '-';
     for (j = 0; j < i; ++j)
          *p++ = c[i - j - 1];

     return (i);
}




/*------------------------------*/
/*   itoROMAN      */
/*------------------------------*/
itoROMAN (value, p, size)
int      value;
register char  *p;
register int  size;
{

/*
 *   convert integer to upper roman. must be positive
 */

     register int  i;
     register int  j;
     register int  k;
     register int  aval;
     char     c[100];
     int      rem;

     aval = abs (value);
     c[0] = EOS;
     i = 1;

     /*
      *   trivial case:
      */
     if (aval == 0)
     {
          c[i++] = '0';
          goto done_100;
     }
```

```
/*
 *   temporarily mod 100...
 */
aval = aval % 100;

if (aval > 0)
{
     /*
      *   build backward
      *
      *   | I|     1
      *   | II|         2
      *   | III|        3
      *   | VI|         4
      *   | V|     5
      *   | IV|         6
      *   | IIV|        7
      *   | IIIV|       8
      *   | XI|         9
      *   | X|     0
      *   | IX|         11
      *   | IIX|        12
      */
     if ((aval % 5 == 0) && (aval % 10 != 0))/* 5 */
          c[i++] = 'V';
     else
     {
          rem = aval % 10;
          if (rem == 9)          /* 9 */
          {
               c[i++] = 'X';
               c[i++] = 'I';
          }
          else if (rem == 8)     /* 8 */
          {
               c[i++] = 'I';
               c[i++] = 'I';
               c[i++] = 'I';
               c[i++] = 'V';
          }
          else if (rem == 7)     /* 7 */
          {
               c[i++] = 'I';
               c[i++] = 'I';
               c[i++] = 'V';
          }
          else if (rem == 6)     /* 6 */
          {
               c[i++] = 'I';
               c[i++] = 'V';
          }
          else if (rem == 4)     /* 4 */
          {
```

```
                c[i++] = 'V';
                c[i++] = 'I';
            }
            else                /* 3,2,1 */
            {
                for (j = 0; j < rem; j++)
                    c[i++] = 'I';
            }
        }

        aval /= 10;
        if (aval == 0)
            goto done_100;

        rem = aval % 10;
        if (rem == 4)
        {
            c[i++] = 'L';
            c[i++] = 'X';
        }
        else if (rem == 5)
        {
            c[i++] = 'L';
        }
        else if (rem < 4)
        {
            for (j = 0; j < rem; j++)
                c[i++] = 'X';
        }
        else
        {
            for (j = 0; j < rem - 5; j++)
                c[i++] = 'X';
            c[i++] = 'L';
        }
    }


done_100:
    /*
     *   divide by 100 (they are done) and temp mod by another 10
     */
    aval  = abs (value);
    aval /= 100;

    if (aval > 0)
    {
        rem  = aval % 10;
        if (rem == 4)
        {
            c[i++] = 'D';
            c[i++] = 'C';
        }
```

```
          if (rem == 5)
          {
               c[i++] = 'D';
          }
          else if (rem < 4)
          {
               for (j = 0; j < rem; j++)
                    c[i++] = 'C';
          }
          else if (rem == 9)
          {
               c[i++] = 'M';
               c[i++] = 'C';
          }
          else if (rem < 9)
          {
               for (j = 0; j < rem - 5; j++)
                    c[i++] = 'C';
               c[i++] = 'D';
          }
     }


     aval /= 10;

     if (aval > 0)
     {
          rem  = aval % 10;
          if (rem < 4)
          {
               for (j = 0; j < rem; j++)
                    c[i++] = 'M';
          }
     }


     if (value < 0)
          c[i++] = '-';

     for (j = 0; j < i; ++j)
          *p++ = c[i - j - 1];

     return (i);
}




/*-----------------------------*/
/*   itoroman      */
/*-----------------------------*/
itoroman (value, p, size)
int     value;
```

```
char   *p;
int     size;
{

/*
 *   convert integer to lower roman
 */

     register int  i;
     register int  len;
     register int  aval;
     char     c[100];

     c[0] = EOS;
     len = itoROMAN (value, c, size);

     for (i = 0; i < len; i++)
     {
          p[i] = c[i];
          if (isalpha (p[i]))
               p[i] = tolower (c[i]);
     }

     return (len);
}




/*-----------------------------*/
/*   itoLETTER     */
/*-----------------------------*/
itoLETTER (value, p, size)
int      value;
register char  *p;
register int  size;
{

/*
 *   convert integer to upper letter value: 0,A,B,C,...,AA,AB,AC,...
 */

     register int  i;
     register int  j;
     register int  k;
     register int  aval;
     int      rem;
     char     c[20];

     aval = abs (value);
     c[0] = EOS;
     i = 1;
```

```
    /*
     *   1 based:
     *
     *   0   0
     *   1   A
     *   25  Z
     *   26  AA
     *   51 AZ
     *   52 AAA
     *   ...
     */
    if (aval == 0)
        c[i++] = '0';
    else if (aval < 27)
    {
        c[i++] = aval - 1 + 'A';
    }
    else
    {
        do
        {
            c[i++] = ((aval - 1) % 26) + 'A';
            aval = (aval - 1)  / 26;

        } while (aval > 0 && i <= size);
    }

    if (value < 0 && i <= size)
        c[i++] = '-';

    for (j = 0; j < i; ++j)
        *p++ = c[i - j - 1];

    return (i);
}



/*------------------------------*/
/*  itoletter     */
/*------------------------------*/
itoletter (value, p, size)
int      value;
register char  *p;
register int  size;
{

/*
 *   convert integer to upper letter value: 0,a,b,c,...,aa,ab,ac,...
 */

    register int  i;
    register int  j;
```

```
    register int  k;
    register int  aval;
    char     c[20];
    int      rem;

    aval = abs (value);
    c[0] = EOS;
    i = 1;

    /*
     *   1 based:
     *
     *   0   0
     *   1   A
     *   25  Z
     *   26  AA
     *   51 AZ
     *   52 AAA
     *   ...
     */
    if (aval == 0)
        c[i++] = '0';
    else if (aval < 27)
    {
        c[i++] = aval - 1 + 'a';
    }
    else
    {
        do
        {
            c[i++] = ((aval - 1) % 26) + 'a';
            aval = (aval - 1)  / 26;

        } while (aval > 0 && i <= size);
    }

    if (value < 0 && i <= size)
        c[i++] = '-';

    for (j = 0; j < i; ++j)
        *p++ = c[i - j - 1];

    return (i);
}



/*------------------------------*/
/*   min           */
/*------------------------------*/

#ifdef min
#undef min
```

```
#endif

min (v1, v2)
register int  v1;
register int  v2;
{

/*
 *   find minimum of two integer ONLY
 */

    return ((v1 < v2) ? v1 : v2);
}




/*------------------------------*/
/*   max           */
/*------------------------------*/

#ifdef max
#undef max
#endif

max (v1, v2)
register int  v1;
register int  v2;
{

/*
 *   find maximum of two integers ONLY
 */

    return ((v1 > v2) ? v1 : v2);
}




/*------------------------------*/
/*   err_exit      */
/*------------------------------*/
#ifdef ALCYON
#include <osbind.h>
#endif

err_exit (code)
{

/*
```

```
 *   exit cleanly on fatal error (close files, etc). also handles normal
 *   exit.
 */

    if (err_stream != stderr && err_stream != (FILE *) 0)
         fclose (err_stream);
    if (dbg_stream != stderr && dbg_stream != (FILE *) 0)
         fclose (dbg_stream);

#ifdef ALCYON
     if (hold_screen)
     {
          printf ("enter any key...");
          Cconin ();
     }
#endif

     exit (code);
}




@//E*O*F low.c//
chmod u=rw,g=r,o=r low.c

echo x - macros.c
sed 's/^@//' > "macros.c" <<'@//E*O*F macros.c//'
/*
 *   macros.c - macro input/output processing for nroff word processor
 *
 *   adapted for atariST/TOS by Bill Rosenkranz 11/89
 *   net: rosenkra@hall.cray.com
 *   CIS: 71460,17
 *   GENIE:    W.ROSENKRANZ
 *
 *   original author:
 *
 *   Stephen L. Browning
 *   5723 North Parker Avenue
 *   Indianapolis, Indiana 46220
 *
 *   history:
 *
 *   - Originally written in BDS C;
 *   - Adapted for standard C by W. N. Paul
 *   - Heavily hacked up to conform to "real" nroff by Bill Rosenkranz
 */

#undef NRO_MAIN                      /* extern globals */

#include <stdio.h>
#include "nroff.h"
```

```
/*------------------------------*/
/*   defmac             */
/*------------------------------*/
defmac (p, infp)
register char  *p;
FILE       *infp;
{

/*
 *   Define a macro. top level, read from stream.
 *
 *   we should read macro without interpretation EXCEPT:
 *
 *   1) number registers are interpolated
 *   2) strings indicated by \* are interpolated
 *   3) arguments indicated by \$ are interpolated
 *   4) concealed newlines indicated by \(newline) are eliminated
 *   5) comments indicated by \" are eliminated
 *   6) \t and \a are interpreted as ASCII h tab and SOH.
 *   7) \\ is interpreted as backslash and \. is interpreted as a period.
 *
 *   currently, we do only 3. a good place to do it would be here before
 *   putmac, after colmac...
 */

    register char  *q;
    register int  i;
    char     name[MNLEN];
    char     defn[MXMLEN];
    char     newend[10];


    /*
     *   skip the .de and get to the name...
     */
    q = skipwd (p);
    q = skipbl (q);

    /*
     *   ok, name now holds the name. make sure it is valid (i.e. first
     *   char is alpha...). getwrd returns the length of the word.
     */
    i = getwrd (q, name);
    if (!isalpha (*name))
    {
         fprintf (err_stream,
              "***%s: missing or illegal macro definition name\n",
              myname);
         err_exit (-1);
    }
```

```
/*
 *   truncate to 2 char max name.
 */
if (i > 2)
     name[2] = EOS;


/*
 *   skip the name and see if we have a new end defined...
 */
q = skipwd (p);
q = skipbl (q);
for (i = 0; i < 10; i++)
     newend[i] = EOS;

for (i = 0; (i < 10) && ( isalpha (q[i]) || isdigit (q[i]) ); i++)
{
     newend[i] = q[i];
}



/*
 *   read a line from input stream until we get the end of macro
 *   command (.en or ..). actually. we should have read the next
 *   field just above here to get the .de NA . or .de NA en string
 *   to be new end of macro.
 */
i = 0;
while (getlin (p, infp) != EOF)
{
     if (p[0] == dc.cmdchr && newend[0] != EOS
     &&  p[1] == newend[0] && p[2] == newend[1])
     {
          /*
           *   replacement end found
           */
          break;
     }
     if (p[0] == dc.cmdchr && p[1] == 'e' && p[2] == 'n')
     {
          /*
           *   .en found
           */
          break;
     }
     if (p[0] == dc.cmdchr && p[1] == dc.cmdchr)
     {
          /*
           *   .. found
           */
          break;
```

```
          }


          /*
           *   collect macro from the line we just read. all this does
           *   is put it in the string defn.
           */
          if ((i = colmac (p, defn, i)) == ERR)
          {
               fprintf (err_stream,
                    "***%s: macro definition too long\n", myname);
               err_exit (-1);
          }
     }


     /*
      *   store the macro
      */
     if (putmac (name, defn) == ERR)
     {
          fprintf (err_stream,
               "***%s: macro definition table full\n", myname);
          err_exit (-1);
     }
}




/*------------------------------*/
/*   colmac             */
/*------------------------------*/
colmac (p, d, i)
register char  *p;
char     d[];
register int  i;
{

/*
 *   Collect macro definition from input stream
 */

     while (*p != EOS)
     {
          if (i >= MXMLEN - 1)
          {
               d[i - 1] = EOS;
               return (ERR);
          }
          d[i++] = *p++;
     }
```

```
    d[i] = EOS;
    return (i);
}




/*-----------------------------*/
/*   putmac             */
/*-----------------------------*/
putmac (name, p)
char   *name;
char   *p;
{

/*
 *   Put macro definition into table
 *
 *   NOTE: any expansions of things like number registers SHOULD
 *   have been done already.
 */


    /*
     *   any room left? (did we exceed max number of possible macros)
     */
    if (mac.lastp >= MXMDEF)
        return (ERR);

    /*
     *   will new one fit in big buffer?
     */
    if (mac.emb + strlen (name) + strlen (p) + 1 > &mac.mb[MACBUF])
    {
        return (ERR);
    }


    /*
     *   add it...
     *
     *   bump counter, set ptr to name, copy name, copy def.
     *   finally increment end of macro buffer ptr (emb).
     *
     *   macro looks like this in mb:
     *
     *   mac.mb[MACBUF]         size of total buf
     *   lastp < MXMDEF         number of macros possible
     *   *mnames[MXMDEF]        -> names, each max length
     *   ..._____________________________..._____________________...
     *       / / /|X|X|0|macro definition      |0| / / / / / / /
     *   .../_/_/_|_|_|_|_________________...___|_|/_/_/_/_/_/_/_...
```

```
     *         ^
     *         |
     *         \----- mac.mnames[mac.lastp] points here
     *
     *   both the 2 char name (XX) and the descripton are null term and
     *   follow one after the other.
     */
    ++mac.lastp;
    mac.mnames[mac.lastp] = mac.emb;
    strcpy (mac.emb, name);
    strcpy (mac.emb + strlen (name) + 1, p);
    mac.emb += strlen (name) + strlen (p) + 2;

    return (OK);
}




/*-----------------------------*/
/*   getmac             */
/*-----------------------------*/
char   *getmac (name)
register char  *name;
{

/*
 *   Get (lookup) macro definition from namespace
 */

    register int  i;

    /*
     *   loop for all macros, starting with last one
     */
    for (i = mac.lastp; i >= 0; --i)
    {
        /*
         *   is this REALLY a macro?
         */
        if (mac.mnames[i])
        {
            /*
             *   if it compares, return a ptr to it
             */
            if (!strcmp (name, mac.mnames[i]))
            {
/*!!!debug             puts (mac.mnames[i]);*/

                if (mac.mnames[i][1] == EOS)
                    return (mac.mnames[i] + 2);
```

```
                else
                    return (mac.mnames[i] + 3);
            }
        }
    }

    /*
     *   none found, return null
     */
    return (NULL);
}




/*------------------------------*/
/*   maceval            */
/*------------------------------*/
maceval (p, m)
register char  *p;
char     m[];
{

/*
 *   Evaluate macro expansion
 */

    register int  i;
    register int  j;
    char       *argp[10];
    char     c;



    /*
     *   replace command char with EOS
     */
    *p++ = EOS;


    /*
     *   initialize argp array to substitute command
     *   string for any undefined argument
     *
     *   NO!!! this is fixed...
     */
    for (i = 0; i < 10; ++i)
        argp[i] = p;

    /*
     *   skip the command name
```

```
 */
p = skipwd (p);
*p++ = EOS;


/*
 *   loop for all $n variables...
 */
for (i = 0; i < 10; ++i)
{
     /*
      *   get to substituted param and if no more, reset remaining
      *   args to NULL and stop...
      */
     p = skipbl (p);
     if (*p == '\r' || *p == '\n' || *p == EOS)
     {
          for ( ; i < 10; i++)
               *argp[i] = '\0';
          break;
     }


     /*
      *   ...otherwise, see if this param is quoted. if it is,
      *   it is all one parameter, even with blanks (but not
      *   newlines...). look for another "c" (which is the quote).
      *
      *   if no quote, just read the arg as a single word and null
      *   terminate it.
      */
     if (*p == '\'' || *p == '"')
     {
          c = *p++;
          argp[i] = p;
          while (*p != c && *p != '\r' && *p != '\n' && *p != EOS)
               ++p;
          *p++ = EOS;
     }
     else
     {
          argp[i] = p;
          p = skipwd (p);
          *p++ = EOS;
     }
}


/*
 *   m contains text of the macro. p contained the input line.
 *   here we start at the end of the macro def and see if there
 *   are any $n thingies. go backwards.
 */
```

```
    for (i = strlen (m) - 1; i >= 0; --i)
    {
        /*
         *   found a $.
         */
        if (i > 0 && m[i - 1] == '$')
        {
            if (!isdigit (m[i]))
            {
                /*
                 *   it wasn't a numeric replacement arg so
                 *   push this char back onto input stream
                 */
                putbak (m[i]);
            }
            else
            {
                /*
                 *   it WAS a numeric replacement arg. so we
                 *   want to push back the appropriate macro
                 *   invocation arg. m[i]-'0' is the numerical
                 *   value of the $0 thru $9. if the arg is
                 *   not there, argp[n] will be (char *) 0
                 *   and pbstr will do nothing.
                 */
                pbstr (argp[m[i] - '0']);
                --i;
            }
        }
        else
        {
            /*
             *   no $ so push back the char...
             */
            putbak (m[i]);
        }
    }

    /*
     *   at this point, the iobuf will hold the new macro command, full
     *   expanded for $n things. the return gets us right back to the
     *   main loop in main() and we parse the (new) command just as if
     *   it were read from a file.
     */

}




/*------------------------------*/
/*   printmac      */
```

```
/*------------------------------*/
printmac (opt)
int  opt;               /* 0=name&size,1=total size,2=full */
{

/*
 *   print all macros and strings and tabulate sizes
 */

     register long i;
     register long space;
     register long totalspace;
     register char  *pname;
     register char  *pdef;


     space      = 0L;
     totalspace = 0L;

     fflush (pout);
     fflush (err_stream);

     for (i = (long) mac.lastp; i >= 0; --i)
     {
          /*
           *   is this REALLY a macro?
           */
          if (mac.mnames[i])
          {
               pname = (char *) (mac.mnames[i]);
               pdef  = pname + 3;
               if (*(pname + 1) == '\0')
                    pdef = pname + 2;

               space       = (long) strlen (pdef);
               totalspace += space;

               switch (opt)
               {
               case 0:
                    fprintf (err_stream, "%s %ld\n", pname, space);
                    break;
               case 2:
                    fprintf (err_stream, "%s %ld\n", pname, space);
                    fprintf (err_stream, "%s\n", pdef);
                    break;
               case 1:
               default:
                    break;
               }
          }
     }
     fprintf (err_stream, "Total space: %ld\n", totalspace);
```

```
}

@//E*O*F macros.c//
chmod u=rw,g=r,o=r macros.c

echo x - strings.c
sed 's/^@//' > "strings.c" <<'@//E*O*F strings.c//'
/*
 *   strings.c - String input/output processing for nroff word processor
 *
 *   adapted for atariST/TOS by Bill Rosenkranz 11/89
 *   net: rosenkra@hall.cray.com
 *   CIS: 71460,17
 *   GENIE:    W.ROSENKRANZ
 *
 *   original author:
 *
 *   Stephen L. Browning
 *   5723 North Parker Avenue
 *   Indianapolis, Indiana 46220
 *
 *   history:
 *
 *   - Originally written in BDS C;
 *   - Adapted for standard C by W. N. Paul
 *   - Heavily hacked up to conform to "real" nroff by Bill Rosenkranz
 */

#undef NRO_MAIN                       /* extern globals */

#include <stdio.h>
#include "nroff.h"



/*------------------------------*/
/*   defstr             */
/*------------------------------*/
defstr (p)
register char  *p;
{

/*
 *   Define a string. top level, read from command line.
 *
 *   we should read string without interpretation EXCEPT:
 *
 *   1) number registers are interpolated
 *   2) strings indicated by \* are interpolated
 *   3) arguments indicated by \$ are interpolated
 *   4) concealed newlines indicated by \(newline) are eliminated
 *   5) comments indicated by \" are eliminated
```

```
*   6) \t and \a are interpreted as ASCII h tab and SOH.
*   7) \\ is interpreted as backslash and \. is interpreted as a period.
*
*   currently, we do only 3. a good place to do it would be here before
*   putstr, after colstr...
*/

    register char  *q;
    register int  i;
    char     name[MNLEN];
    char     defn[MXMLEN];



    name[0] = '\0';
    defn[0] = '\0';


    /*
     *   skip the .ds and get to the name...
     */
    q = skipwd (p);
    q = skipbl (q);

    /*
     *   ok, name now holds the name. make sure it is valid (i.e. first
     *   char is alpha...). getwrd returns the length of the word.
     */
    i = getwrd (q, name);
    if (!name[0])
    {
         fprintf (err_stream,
              "***%s: missing or illegal string definition name\n",
              myname);
         err_exit (-1);
    }

    /*
     *   truncate to 2 char max name.
     */
    if (i > 2)
         name[2] = EOS;


    /*
     *   skip the name to get to the string. it CAN start with a " to
     *   have leading blanks...
     */
    q = skipwd (q);
    q = skipbl (q);
```

```
     /*
      *   read rest of line from input stream and collect string into
      *   temp buffer defn
      */
     if ((i = colstr (q, defn)) == ERR)
     {
          fprintf (err_stream,
               "***%s: string definition too long\n", myname);
          err_exit (-1);
     }


     /*
      *   store the string
      */
     if (putstr (name, defn) == ERR)
     {
          fprintf (err_stream,
               "***%s: string definition table full\n", myname);
          err_exit (-1);
     }
}




/*-----------------------------*/
/*   colstr             */
/*-----------------------------*/
colstr (p, d)
register char  *p;
char     d[];
{

/*
 *   Collect string definition from input stream
 */

     register int  i = 0;

     if (*p == '\"')
          p++;

     while (*p != EOS)
     {
          if (i >= MXMLEN - 1)
          {
               d[i - 1] = EOS;
               return (ERR);
          }
          d[i++] = *p++;
     }
```

```
    d[i] = EOS;
    return (i);
}




/*------------------------------*/
/*   putstr             */
/*------------------------------*/
putstr (name, p)
register char  *name;
register char  *p;
{

/*
 *   Put string definition into (macro) table
 *
 *   NOTE: any expansions of things like number registers SHOULD
 *   have been done already. strings and macros share mb buffer
 */


    /*
     *   any room left? (did we exceed max number of possible macros)
     */
    if (mac.lastp >= MXMDEF)
        return (ERR);

    /*
     *   will new one fit in big buffer?
     */
    if (mac.emb + strlen (name) + strlen (p) + 1 > &mac.mb[MACBUF])
    {
        return (ERR);
    }


    /*
     *   add it...
     *
     *   bump counter, set ptr to name, copy name, copy def.
     *   finally increment end of macro buffer ptr (emb).
     *
     *   string looks like this in mb:
     *
     *   mac.mb[MACBUF]         size of total buf
     *   lastp < MXMDEF         number of macros/strings possible
     *   *mnames[MXMDEF]        -> names, each max length
     *   ..._____________________________..._____________________...
     *       / / /|X|X|0|string definition      |0| / / / / / / /
     *   .../_/_/_|_|_|_|_________________...___|_|/_/_/_/_/_/_/_...
```

```
     *           ^
     *           |
     *           \----- mac.mnames[mac.lastp] points here
     *
     *   both the 2 char name (XX) and the descripton are null term and
     *   follow one after the other.
     */
    ++mac.lastp;
    mac.mnames[mac.lastp] = mac.emb;
    strcpy (mac.emb, name);
    strcpy (mac.emb + strlen (name) + 1, p);
    mac.emb += strlen (name) + strlen (p) + 2;
    return (OK);
}




/*------------------------------*/
/*   getstr             */
/*------------------------------*/
char   *getstr (name)
register char  *name;
{

/*
 *   Get (lookup) string definition from namespace
 */

    register int  i;

    /*
     *   loop for all macros, starting with last one
     */
    for (i = mac.lastp; i >= 0; --i)
    {
        /*
         *   is this REALLY a macro?
         */
        if (mac.mnames[i])
        {
            /*
             *   if it compares, return a ptr to it
             */
            if (!strcmp (name, mac.mnames[i]))
            {
/*!!!debug            puts (mac.mnames[i]);*/

                if (mac.mnames[i][1] == EOS)
                    return (mac.mnames[i] + 2);
                else
```

```
                        return (mac.mnames[i] + 3);
                }
            }
        }

        /*
         *   none found, return null
         */
        return (NULL);
}




@//E*O*F strings.c//
chmod u=rw,g=r,o=r strings.c

exit 0


From -4580591408906204717
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,d4408e088a0897a3,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-11-14 20:50:01 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!mailrus!wuarchive!texbell!texsun!
newstop!sun!sunfedcomm!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i011: kbext410 -- Keyboard remapper
Keywords: shar, 68000
Message-ID: <34767@grapevine.uucp>
Date: 15 Nov 89 04:50:01 GMT
Sender: daemon@sunfedcomm
Reply-To: BD05@DKAUNI2.Bitnet (Roland Waldi)
Lines: 837
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: BD05@DKAUNI2.Bitnet (Roland Waldi)
Posting-number: Volume 3, Issue 11
Archive-name: kbext410

This is the latest version (4.10) of KBEXT (keyboard extension).
Put KBEXT4.PRG into the AUTO folder, or start it at any time from the
desktop (but only once a session). It will establish a memory resident
extension to the keyboard interrupt handler, which allows in addition to
the NORMAL, SHIFT and CAPS-LOCK keyboard tables also ALT and ALT-SHIFT
keyboard tables to be used, thus allowing more than 128 additional
characters to be entered directly from the keyboard pressing the ALT
key. ALL KEYBOARD TABLES (INCL. THE NON-ALT) OVERRIDE ANY OTHER USER- OR
PROGRAM-DEFINED TABLE, I.E. XBIOS CALL 16 (KEYTBL) HAS NO EFFECT! COLD
```

BOOT is triggered by pressing CTRL-ALT-rightSHIFT-DEL, WARM BOOT by
pressing CTRL-ALT-DEL (it's interesting to hear TOS 1.4 has this already
built-in :-)).
KBEXT4 contains also a screen saver, which disables the screen 10
minutes after the last mouse/keyboard/midi event; KBEXT3 is without
screen saver.
The program may be freely used, copied and modified, totally at your own
risk and responsibility, and only for non-profit use.

Enjoy,   Roland Waldi, BD05 @ DKAUNI2.BITNET

```
#!/bin/sh
# shar:  Shell Archiver  (v1.22)
#
#    Run the following text with /bin/sh to create:
#      KBEXT.TXT
#      KBEXT3.ASM
#      KBEXT4.ASM
#
sed 's/^X//' << 'SHAR_EOF' > KBEXT.TXT &&
XKBEXT.PRG (Version 2.02-87/09, 3.10-89/11, 4.10-89/11)
X-------------------------------------------------------
X
XPut KBEXT.PRG into the AUTO folder, or start it at any time from the
Xdesktop (but only once a session). It will establish a memory
Xresident extension to the keyboard interrupt handler, which allows
Xin addition to the NORMAL, SHIFT and CAPS-LOCK keyboard tables also
XALT and ALT-SHIFT keyboard tables to be used, thus allowing more than
X128 additional characters to be entered directly from the keyboard
Xpressing the ALT key. For use with some programs using ALT+key
Xcombinations as control keys, use ALT+CTRL+key instead.
X
XKBEXT version 3.xx does the same, but all keyboard tables (incl. the
Xnon-ALT) override any other user- or program-defined table, i.e. XBIOS
Xcall 16 (keytbl) has no effect! The version 2.xx only initializes the
XTOS keyboard table to its own copies, using keytbl itself, which may be
Xreplaced or abandoned later by any other program.
X
XKBEXT version 4.xx is KBEXT 3.xx plus a screen saver, which disables
Xthe screen 10 minutes after the last mouse/keyboard/midi event.
XKBEXT version 4.02 adds a cold boot by pressing CTRL-ALT-rightSHIFT-DEL.
XKBEXT version 4.10 adds a warm boot by pressing CTRL-ALT-DEL.
X
XIf you use KBEXT.PRG as it is, you will get the following keyboard
Xlayout (which is close to the US and GB version of the ATARI keyboard
Xfor the non-ALT keys):
X
XNORMAL:
X
XF1 /F2 /F3 /F4 /F5 /F6 /F7 /F8 /F9 /F10/
X
XEsc 1 2 3 4 5 6 7 8 9 0 - = `   Bsp  Help     Undo   (   )  /    *
XTab  q w e r t y u i o p [ ]    Del  Insrt Up  Clr   7   8  9    -
```

```
XCtrl  a s d f g h j k l ; '  Ret  #  Left Dwn Rght   4   5  6    +
XShft \ z x c v b n m , . /  Shft                    1   2  3  Ent
X    Alt  -----Space-----  CpsL                       0    .
X
XSHIFT (note that no digits appear with the cursor keys, which
Xmeans you need not release the SHIFT key to move the cursor!)
X
XF11/F12/F13/F14/F15/F16/F17/F18/F19/F20/
X
XEsc ! @ # $ % ^ & * ( ) _ + :   Bsp  Help     Undo   (   )  /    *
XTab  Q W E R T Y U I O P { }    Del  Insrt Up  Clr   7   8  9    -
XCtrl  A S D F G H J K L : "  Ret  ~  Left Dwn Rght   4   5  6    +
XShft | Z X C V B N M < > ?  Shft                    1   2  3  Ent
X    Alt  -----Space-----  CpsL                       0    .
X
XCAPS-LOCK:
X
XF1 /F2 /F3 /F4 /F5 /F6 /F7 /F8 /F9 /F10/
X
XEsc 1 2 3 4 5 6 7 8 9 0 - = `   Bsp  Help     Undo   (   )  /    *
XTab  Q W E R T Y U I O P [ ]    Del  Insrt Up  Clr   7   8  9    -
XCtrl  A S D F G H J K L ; '  Ret  #  Left Dwn Rght   4   5  6    +
XShft \ Z X C V B N M , . /  Shft                    1   2  3  Ent
X    Alt  -----Space-----  CpsL                       0    .
X
XALT:
X
XF1 /F2 /F3 /F4 /F5 /F6 /F7 /F8 /F9 /F10/
X
XEsc ; +  , ? ) = y x 2  p K   Bsp  SDump          `   a  m    o
XTab   B  U g
  c G u    Del  LMous MUp RMs   !

XCtrl    k  D F @ L M H J  Ret    MLeft MDn MRgt           X
XShft 4 0 1  ^ C $ f [ \ 3  Shft
  Ent
X    Alt  -----Space-----  CpsL                       i    e
X
XSHIFT & ALT:
X
XF11/F12/F13/F14/F15/F16/F17/F18/F19/F20/
X
XEsc - 9 ]  v * > z & ' w q    Bsp  SDump          .   /  h    6
XTab   V n T S   l  < O t    Del  LMous Mup RMs   "         Z
XCtrl   d E N b I A Q R S T  Ret    Mleft Mdn Mrgt  #         W
XShft 5 7 8  { _ % P s r (  Shft                       }  ~  Ent
X    Alt  -----Space-----  CpsL                       j    |
X
XThere are at least two ways to change the keyboard assignment:
X
X1. Change the keyboard tables in KBEXT.ASM and reassemble the
X   program. You may have to change some control statements
X   to fit your assembler's requirements, but the program is
```

```
X   written to make changes even to MEGAMAX-C-Inline-Assembler
X   not too hard (Use a minimal INIT.C here to save memory and
X   to provide basepage information).
X
X2. Use any PD keyboard-changing program, which allows to
X   save a xxx.KBD file (check if it has 384 bytes), e.g.
X   MOBZKEY.ACC, and make two files: DEFAULT.KBD for the
X   normal keyboard assignment, and ALT.KBD with the
X   unshifted and shifted ALT keyboard assignments (don't
X   change the caps-lock table here, you will need CAPS LOCK
X   to type into the file-selector box, since MOBZKEY changes
X   the keyboard assignment immediately!).
X
X   Put these two files into the \AUTO\ folder of your boot disk,
X   if you use KBEXT at boot time, or into the \AUTO\ folder of
X   your working disk containig KBEXT(4).PRG otherwise.
X   From now on the tables in these datasets will be used, when KBEXT
X   is run.
X
X3. A third possibility would be to make the files DEFAULT.KBD and
X   ALT.KBD mentioned in (2) by a separate program, e.g. a file monitor
X   etc. The format of a KBD file is simply 3 times 128 bytes of the
X   keyboard tables (one byte per character) for normal, shifted and
X   caps-lock. The key numbers (hexadecimal) are:
X
X3B /3C /3D /3E /3F /40 /41 /42 /43 /44 /
X
X01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 29  0E  62    61  63 64 65 66
X0F  10 11 12 13 14 15 16 17 18 19 1A 1B    53  52 48 47  67 68 69 4A
X[1D] 1E 1F 20 21 22 23 24 25 26 27 28   1C 2B  4B 50 4D  6A 6B 6C 4E
X[2A](60)2C 2D 2E 2F 30 31 32 33 34 35 [36]               6D 6E 6F 72
X   [38]  -------39------  [3A]                           70   71
X
X   Note that US keyboards don't have the key (60). You should assign
X   0 to all function keys, and to the positions that do not belong
X   to any key. Key number redefinitions used by TOS with the ALT key are
X   automatically taken into account by KBEXT, after reading the ALT.KBD
X   file; so the numbers given above should be used for all 5 tables.
X
X4. Finally, you can also patch the program code: you find all tables
X   contiguous after "KBEXT/RW" in the order normal - shift - caps -
X   alt - shift/alt. You can use your favourite disk-/file monitor
X   to change these tables in KBEXT(4).PRG.
X
XThe program may be freely used, copied and modified, totally at your
Xown risk and responsibility, and only for non-profit use.
X
XEnjoy,                Roland Waldi, BD05 @ DKAUNI2.BITNET
SHAR_EOF
chmod 0600 KBEXT.TXT || echo "restore of KBEXT.TXT fails"
sed 's/^X//' << 'SHAR_EOF' > KBEXT3.ASM &&
X;************************************************************
X;*                                                          *
```

```
X;* KBEXT3.ASM: extend keyboard functions                        *
X;* installs itself in the keyboard interrupt                    *
X;* valid for TOS 1.00 20/11/1985 or TOS 1.2 and following      *
X;*                                                             *
X;* TOS version dependent (marked with **):                     *
X;*     addr. for kbshifts: A5D (TOS 20.06.85 D)                *
X;*                         E1B (TOS 20.11.85 GB)               *
X;*                         E1B (TOS 06.02.86 D)                *
X;*     for TOS 1.2 ff. automatically set                       *
X;*                                                             *
X;* THIS VERSION (3.xx) IGNORES ALL TOS KEYBOARD TABLES         *
X;* HAS A WARMBOOT on Ctrl-Alt-Del                              *
X;* HAS A COLDBOOT on Ctrl-Alt-rightShift-Del                   *
X;* Version for GST-Assembler                                   *
X;*                                                             *
X;***** (C) Roland Waldi *** 3.10-89/11 ************************
X
X          SECTION   MAIN
X          RORG      0
X          BRA       install
X
Xkbint:
X          MOVE.L    (A7)+,RTNadd   ;* return address
X          MOVE.L    TOSvec,A0
X          JSR       (A0)           ;* jump to TOS routine
X          MOVE.L    RTNadd,-(A7)   ;* return address
X          BRA.S     go
X
Xtimerint:
X          MOVE.L    (A7)+,RTNadd   ;* return address
X          MOVE.L    TMRvec,A0
X          JSR       (A0)           ;* call etv_timer
X          MOVE.L    RTNadd,-(A7)   ;* return address
Xgo:
X          MOVE.L    IOrec,A0
X          MOVE.L    (A0),A2        ;* get address of io-buffer
X          MOVE.W    8(A0),D1       ;* tail index
X          CMP.W     OLDtail,D1     ;* test if changed
X          BEQ.S     rturn
X          LEA       OLDtail,A1
X          MOVE.W    D1,(A1)        ;* save tail index at OLDtail
Xkbshadr:  MOVE.B    $E1B,D2        ;* get kbshifts (** TOS dependent **)
X          BTST      #2,D2          ;* CTRL set?
X          BEQ.S     noctrl         ;*   if not, translate
X          BTST      #3,D2          ;* ALT set?
X          BEQ.S     rturn          ;*   if not, return
X          MOVE.W    0(A2,D1.W),D0  ;* get...
X          AND.W     #$7F,D0        ;* ...key code
X          CMP.B     #$53,D0
X          BNE.S     nospecl
X          BTST      #0,D2          ;* right SHIFT set?
X          BNE.S     reset          ; C-A Del: Boot
X          CLR.L     $420           ; C-A-rS Del: Coldboot
```

```
X          CLR.L     $426
X          CLR.L     $43A
Xreset:    RESET
Xnospecl:
X          CLR.W     D0
X          BRA.S     store          ;* store 0
Xnoctrl:
X          MOVE.W    0(A2,D1.W),D0  ;* get...
X          AND.W     #$7F,D0        ;* ...key code
X          BTST      #3,D2          ;* ALT set?
X          BNE.S     altkeys        ;*   if 1
X          LEA       normal,A1      ;* get normal table
X          AND.B     #$13,D2        ;* test SHIFT l/r and CapsL
X          BEQ.S     goon           ;*   if none set
X          ADD.W     #128,D0        ;* use second table (SHIFT)
X          AND.B     #$03,D2        ;* only CapsLock set?
X          BNE.S     goon           ;* if not
X          ADD.W     #128,D0        ;* CapsLock table
X          BRA.S     goon
Xaltkeys:
X          LEA       ALTtabl,A1     ;* get translation table
X          AND.B     #$13,D2        ;* test SHIFT l/r and CapsL
X          BEQ.S     goon           ;* if none set
X          ADD.W     #128,D0        ;* use second table (SHIFT)
Xgoon:
X          MOVE.B    0(A1,D0.W),D0  ;* translate
Xstore:
X          MOVE.W    D0,2(A2,D1.W)  ;* store char code
Xrturn:
X          RTS
XTOSvec:
X          DC.L      0
XTMRvec:
X          DC.L      0
XRTNadd:
X          DC.L      0
XIOrec:
X          DC.L      0
XOLDtail:
X          DC.W      -1
Xtag:
X          DC.B      'K','B','E','X','T','/','R','W'
X;*
X;* here are the keyboard tables for normal, SHIFT,
X;*                                  and normal with CAPS LOCK
X;*                                  ALT, ALT&SHIFT
X;* you find them in the compiled program after the tag "KBEXT/RW"
X;*
X;* to change the keyboard layout, use the "normal" table as a
X;* guide for the keys (the layout is for a UK keyboard)
X;* and insert the ATARI/ASCII codes of the characters you want.
X;* NOTE:
X;* with ALT, the positions 02 ('1') ... 09 ('8') move to
```

```
X;* 78 ... 7F,
X;* and 0A ('9') ... 0D ('=') to 00 ... 03 (should be 80 ... 83)
X;* these rearrangements are done BY THE PROGRAM, so: fill the
X;* tables according to the normal keycode position.
X;*
Xnormal:
X          DC.B      $00,$1B,'1','2','3','4','5','6' ;* 00 Esc
X          DC.B      '7','8','9','0','-','=',$08,$09 ;* 08 BS,TAB
X          DC.B      'q','w','e','r','t','y','u','i' ;* 10
X          DC.B      'o','p',$5B,$5D,$0D,$00,'a','s' ;* 18 brackets,CR
X          DC.B      'd','f','g','h','j','k','l',';' ;* 20
X          DC.B      $27,'`',$00,'#','z','x','c','v' ;* 28 '
X          DC.B      'b','n','m',',','.','/',$00,$00 ;* 30
X          DC.B      $00,' ',$00,$00,$00,$00,$00,$00 ;* 38
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 40
X          DC.B      $00,$00,'-',$00,$00,$00,'+',$00 ;* 48
X          DC.B      $00,$00,$00,$7F,$00,$00,$00,$00 ;* 50 DEL
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 58
X          DC.B      $5C,$00,$00,'(',')','/','*','7' ;* 60 \
X          DC.B      '8','9','4','5','6','1','2','3' ;* 68
X          DC.B      '0','.',$0D,$00,$00,$00,$00,$00 ;* 70 Enter
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 78
Xshiftblk:
X          DC.B      $00,$1B,'!','@','#','$','%','^' ;* 00
X          DC.B      '&','*','(',')','_','+',$08,$09 ;* 08
X          DC.B      'Q','W','E','R','T','Y','U','I' ;* 10
X          DC.B      'O','P','{','}',$0D,$00,'A','S' ;* 18
X          DC.B      'D','F','G','H','J','K','L',':' ;* 20
X          DC.B      '"',$BA,$00,'~','Z','X','C','V' ;* 28
X          DC.B      'B','N','M','<','>','?',$00,$00 ;* 30
X          DC.B      $00,' ',$00,$00,$00,$00,$00,$00 ;* 38
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 40
X          DC.B      $00,$00,'-',$00,$00,$00,'+',$00 ;* 48
X          DC.B      $00,$00,$00,$7F,$00,$00,$00,$00 ;* 50
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 58
X          DC.B      '|',$00,$00,'(',')','/','*','7' ;* 60
X          DC.B      '8','9','4','5','6','1','2','3' ;* 68
X          DC.B      '0','.',$0D,$00,$00,$00,$00,$00 ;* 70
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 78
Xcapslock:
X          DC.B      $00,$1B,'1','2','3','4','5','6' ;* 00
X          DC.B      '7','8','9','0','-','=',$08,$09 ;* 08
X          DC.B      'Q','W','E','R','T','Y','U','I' ;* 10
X          DC.B      'O','P',$5B,$5D,$0D,$00,'A','S' ;* 18
X          DC.B      'D','F','G','H','J','K','L',';' ;* 20
X          DC.B      $27,'`',$00,'#','Z','X','C','V' ;* 28
X          DC.B      'B','N','M',',','.','/',$00,$00 ;* 30
X          DC.B      $00,' ',$00,$00,$00,$00,$00,$00 ;* 38
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 40
X          DC.B      $00,$00,'-',$00,$00,$00,'+',$00 ;* 48
X          DC.B      $00,$00,$00,$7F,$00,$00,$00,$00 ;* 50
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 58
X          DC.B      $5C,$00,$00,'(',')','/','*','7' ;* 60
```

```
X          DC.B      '8','9','4','5','6','1','2','3' ;* 68
X          DC.B      '0','.',$0D,$00,$00,$00,$00,$00 ;* 70
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 78
XALTtabl:
X          DC.B      $00,$1B,$BB,$AB,$9C,$AC,$BF,$A9 ;* 00 (reduced set)
X          DC.B      $BD,$F9,$F8,$B2,$7F,$F0,$00,$00 ;* 08
X          DC.B      $91,$C2,$89,$D5,$E7,$98,$81,$8B ;* 10
X          DC.B      $94,$E3,$00,$F5,$00,$00,$84,$9E ;* 18
X          DC.B      $EB,$9F,$00,$00,$C0,$00,$00,$00 ;* 20
X          DC.B      $00,$00,$00,$86,$B0,$B1,$87,$DE ;* 28
X          DC.B      $00,$A4,$E6,$DB,$DC,$B3,$00,$00 ;* 30
X          DC.B      $00,$20,$00,$00,$00,$00,$00,$00 ;* 38
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 40
X          DC.B      $00,$00,$D9,$00,$00,$00,$D8,$00 ;* 48
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 50
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 58
X          DC.B      $B4,$00,$00,$E0,$E1,$ED,$EF,$A1 ;* 60
X          DC.B      $8C,$8D,$A0,$83,$85,$82,$88,$8A ;* 68
X          DC.B      $E9,$E5,$00,$00,$00,$00,$00,$00 ;* 70
X          DC.B      $BB,$AB,$9C,$AC,$BF,$A9,$BD,$F9 ;* 78
X;* ALT & SHIFT
X          DC.B      $00,$1B,$AD,$B9,$DD,$9B,$F6,$AA ;* 00
X          DC.B      $BE,$FA,$A6,$A7,$F7,$F1,$00,$00 ;* 08
X          DC.B      $92,$D6,$EE,$D4,$D3,$9D,$9A,$EC ;* 10
X          DC.B      $99,$BC,$CF,$F4,$00,$00,$8E,$E4 ;* 18
X          DC.B      $C5,$CE,$E2,$C9,$C1,$D1,$D2,$D3 ;* 20
X          DC.B      $D4,$FF,$00,$8F,$B7,$B8,$80,$FB ;* 28
X          DC.B      $DF,$A5,$D0,$F3,$F2,$A8,$00,$00 ;* 30
X          DC.B      $00,$20,$00,$00,$00,$00,$00,$00 ;* 38
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 40
X          DC.B      $00,$00,$DA,$00,$00,$00,$D7,$00 ;* 48
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 50
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 58
X          DC.B      $B5,$00,$00,$AE,$AF,$E8,$B6,$A2 ;* 60
X          DC.B      $93,$95,$A3,$96,$97,$90,$FD,$FE ;* 68
X          DC.B      $EA,$FC,$00,$00,$00,$00,$00,$00 ;* 70
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 78
X
Xpglength:
X          DC.L      0
Xinstall:
X          MOVEA.L   4(A7),A5       ;* basepage
X          LEA       pglength(PC),A0
X          LEA       install(PC),A1
X          MOVE.L    A1,D0
X          SUB.L     A5,D0          ;* calculate length
X          MOVE.L    D0,(A0)
X          MOVE.L    D0,-(A7)       ;* length
X          MOVE.L    A5,-(A7)       ;* start
X          CLR.W     -(A7)
X          MOVE.W    #$4A,-(A7)
X          TRAP      #1             ;* SETBLOCK, reserve storage
X          LEA       12(A7),A7
```

```
X           LEA       messg1(PC),A0  ;* display message
X           BSR       outp
X           CLR.W     -(A7)          ;* read file DEFAULT.KBD
X           PEA       file1(PC)
X           MOVE.W    #$3D,-(A7)     ;* open
X           TRAP      #1
X           ADDQ.L    #8,A7
X           TST.W     D0
X           BMI.S     nodefault
X           PEA       normal(PC)     ;* table address for read
X           MOVE.L    #384,-(A7)     ;* length
X           MOVE.W    D0,-(A7)       ;* handle
X           MOVE.W    #$3F,-(A7)     ;* read
X           TRAP      #1
X           LEA       12(A7),A7
X           MOVE.W    handle(PC),-(A7)
X           MOVE.W    #$3E,-(A7)     ;* close
X           TRAP      #1
X           ADDQ.L    #4,A7
X           LEA       file1(PC),A0   ;* display file1
X           BSR       outp
Xnodefault:
X           LEA       messg2(PC),A0  ;* display message
X           BSR       outp
X           CLR.W     -(A7)          ;* read file ALT.KBD
X           PEA       file2(PC)
X           MOVE.W    #$3D,-(A7)     ;* open
X           TRAP      #1
X           ADDQ.L    #8,A7
X           TST.W     D0
X           BMI.S     noaltfile
X           LEA       handle(PC),A0  ;* file handle
X           MOVE.W    D0,(A0)
X           PEA       ALTtabl(PC)    ;* table address for read
X           MOVE.L    #256,-(A7)     ;* length
X           MOVE.W    D0,-(A7)       ;* handle
X           MOVE.W    #$3F,-(A7)     ;* read
X           TRAP      #1
X           LEA       12(A7),A7
X           MOVE.W    handle(PC),-(A7)
X           MOVE.W    #$3E,-(A7)     ;* close
X           TRAP      #1
X           ADDQ.L    #4,A7
X           LEA       file2(PC),A0   ;* display file2
X           BSR       outp
Xnoaltfile:
X           LEA       ALTtabl(PC),A0 ;* rearrange ALT-Codes
X           MOVE.L    $02(A0),$78(A0)
X           MOVE.L    $06(A0),$7C(A0)
X           MOVE.L    $0A(A0),$00(A0)
X           MOVE.L    $082(A0),$0F8(A0) ;* same for ALT SHIFT
X           MOVE.L    $086(A0),$0FC(A0)
X           MOVE.L    $08A(A0),$080(A0)
```

```
X           MOVE.W    #1,-(A7)
X           MOVE.W    #14,-(A7)
X           TRAP      #14             ;* get iorec(1)
X           ADDQ.L    #4,A7
X           LEA       IOrec(PC),A0
X           MOVE.L    D0,(A0)         ;* save pointer to kbd-iorec
X           MOVE.W    #34,-(A7)
X           TRAP      #14             ;* Kbdvbase()
X           ADDQ.L    #2,A7
X           MOVE.L    D0,A1           ;* A1 = addr(kbdvbase)
X           LEA       TOSvec,A0
X           MOVE.L    32(A1),(A0)     ;* save interrupt routine address
X           LEA       kbint,A0
X           MOVE.L    A0,32(A1)       ;* install new interrupt vectors
X           CLR.L     -(A7)
X           MOVE.W    #$20,-(A7)
X           TRAP      #1              ;* Super
X           ADDQ.L    #6,A7
X           LEA       TMRvec,A0
X           MOVE.L    $400,(A0)       ;* save etv_timer routine address
X           LEA       timerint,A0
X           MOVE.L    A0,$400         ;* install new interrupt vector
X           MOVE.L    $4F2,A0         ;* get system address
X           CMPI.B    #2,3(A0)        ;* check subversion number
X           BLT.S     oldsys          ;* TOS < 1.2: keep address
X           MOVE.W    26(A0),kbshadr+2  ;* copy kbshift variable address
Xoldsys:
X           MOVE.L    D0,-(A7)
X           MOVE.W    #$20,-(A7)
X           TRAP      #1
X           ADDQ.L    #6,A7
X           LEA       messg3(PC),A0   ;* newline
X           BSR.S     outp
X           CLR.W     -(A7)           ;* return code
X           MOVE.L    pglength(PC),-(A7) ;* program length
X           MOVE.W    #$31,-(A7)      ;* Ptermres
X           TRAP      #1              ;* reserve storage and end
Xoutp:      MOVE.L    A0,-(A7)
X           MOVE.W    #$09,-(A7)
X           TRAP      #1              ;* print line
X           ADDQ.L    #6,A7
X           RTS                       ;* never reached
Xhandle:    DC.W      0
Xmessg1:    DC.B      $0D,$0A,'KBEXT v. 3.10 installing keyboard ',$00
Xmessg2:    DC.B      $0D,$0A,'------------- and ALT table       ',$00
Xmessg3:    DC.B      $0D,$0A,$00
Xfile1:     DC.B      $5C,'AUTO',$5C
X           DC.B      'DEFAULT.KBD',$00
Xfile2:     DC.B      $5C,'AUTO',$5C
X           DC.B      'ALT.KBD',$00
X           END
SHAR_EOF
chmod 0600 KBEXT3.ASM || echo "restore of KBEXT3.ASM fails"
```

```
sed 's/^X//' << 'SHAR_EOF' > KBEXT4.ASM &&
X;*************************************************************
X;*                                                           *
X;* KBEXT4.ASM: extend keyboard functions                     *
X;* installs itself in the keyboard interrupt                 *
X;* valid for TOS 1.00 20/11/1985 or TOS 1.2 and following    *
X;*                                                           *
X;* TOS version dependent (marked with **):                   *
X;*     addr. for kbshifts: A5D (TOS 20.06.85 D)              *
X;*                         E1B (TOS 20.11.85 GB)             *
X;*                         E1B (TOS 06.02.86 D)              *
X;*     for TOS 1.2 ff. automatically set                     *
X;*                                                           *
X;* THIS VERSION (4.xx) IGNORES ALL TOS KEYBOARD TABLES       *
X;* HAS A WARMBOOT on Ctrl-Alt-Del                            *
X;* HAS A COLDBOOT on Ctrl-Alt-rightShift-Del                 *
X;* AND INCLUDES A SCREEN SAVER                               *
X;* Version for GST-Assembler                                 *
X;*                                                           *
X;***** (C) Roland Waldi *** 4.10-89/11 ************************
X
X          SECTION   MAIN
X          RORG      0
X          BRA       install
X
Xacia:
X          TST.W     W2
X          BEQ.S     L9
X          BCLR      #0,$FF820A
X          CLR.W     W2
XL9:       CLR.L     X3
X          MOVE.L    olacia,-(A7)
X          RTS
X
Xkbint:
X          MOVE.L    (A7)+,RTNadd   ;* return address
X          MOVE.L    TOSvec,A0
X          JSR       (A0)           ;* jump to TOS routine
X          MOVE.L    RTNadd,-(A7)   ;* return address
X          BRA.S     go
X
Xtimerint:
X          ADDQ.L    #1,X3
X          CMPI.L    #45000,X3      ;* 15 min
X          BLS.S     L3
XLOOP1N:   CMPI.B    #$80,$FF8207
X          BNE.S     LOOP1N
X          BSET      #0,$FF820A
X          ADDQ.W    #1,W2
XL3:
X          MOVE.L    (A7)+,RTNadd   ;* return address
X          MOVE.L    TMRvec,A0
X          JSR       (A0)           ;* call etv_timer
```

```
X          MOVE.L    RTNadd,-(A7)   ;* return address
Xgo:
X          MOVE.L    IOrec,A0
X          MOVE.L    (A0),A2        ;* get address of io-buffer
X          MOVE.W    8(A0),D1       ;* tail index
X          CMP.W     OLDtail,D1     ;* test if changed
X          BEQ.S     rturn
X          LEA       OLDtail,A1
X          MOVE.W    D1,(A1)        ;* save tail index at OLDtail
Xkbshadr:  MOVE.B    $E1B,D2        ;* get kbshifts (** TOS dependent **)
X          BTST      #2,D2          ;* CTRL set?
X          BEQ.S     noctrl         ;*   if not, translate
X          BTST      #3,D2          ;* ALT set?
X          BEQ.S     rturn          ;*   if not, return
X          MOVE.W    0(A2,D1.W),D0  ;* get...
X          AND.W     #$7F,D0        ;* ...key code
X          CMP.B     #$53,D0
X          BNE.S     nospecl
X          BTST      #0,D2          ;* right SHIFT set?
X          BNE.S     reset          ; C-A Del: Boot
X          CLR.L     $420           ; C-A-rS Del: Coldboot
X          CLR.L     $426
X          CLR.L     $43A
Xreset:    RESET
Xnospecl:
X          CLR.W     D0
X          BRA.S     store          ;* store 0
Xnoctrl:
X          MOVE.W    0(A2,D1.W),D0  ;* get...
X          AND.W     #$7F,D0        ;* ...key code
X          BTST      #3,D2          ;* ALT set?
X          BNE.S     altkeys        ;*   if 1
X          LEA       normal,A1      ;* get normal table
X          AND.B     #$13,D2        ;* test SHIFT l/r and CapsL
X          BEQ.S     goon           ;*   if none set
X          ADD.W     #128,D0        ;* use second table (SHIFT)
X          AND.B     #$03,D2        ;* only CapsLock set?
X          BNE.S     goon           ;* if not
X          ADD.W     #128,D0        ;* CapsLock table
X          BRA.S     goon
Xaltkeys:
X          LEA       ALTtabl,A1     ;* get translation table
X          AND.B     #$13,D2        ;* test SHIFT l/r and CapsL
X          BEQ.S     goon           ;* if none set
X          ADD.W     #128,D0        ;* use second table (SHIFT)
Xgoon:
X          MOVE.B    0(A1,D0.W),D0  ;* translate
Xstore:
X          MOVE.W    D0,2(A2,D1.W)  ;* store char code
Xrturn:
X          RTS
XTOSvec:
X          DC.L      0
```

```
XTMRvec:
X          DC.L      0
XRTNadd:
X          DC.L      0
XIOrec:
X          DC.L      0
XOLDtail:
X          DC.W      -1
Xtag:
X          DC.B      'K','B','E','X','T','/','R','W'
X;*
X;* here are the keyboard tables for normal, SHIFT,
X;*                                   and normal with CAPS LOCK
X;*                                   ALT, ALT&SHIFT
X;* you find them in the compiled program after the tag "KBEXT/RW"
X;*
X;* to change the keyboard layout, use the "normal" table as a
X;* guide for the keys (the layout is for a UK keyboard)
X;* and insert the ATARI/ASCII codes of the characters you want.
X;* NOTE:
X;* with ALT, the positions 02 ('1') ... 09 ('8') move to
X;* 78 ... 7F,
X;* and 0A ('9') ... 0D ('=') to 00 ... 03 (should be 80 ... 83)
X;* these rearrangements are done BY THE PROGRAM, so: fill the
X;* tables according to the normal keycode position.
X;*
Xnormal:
X          DC.B      $00,$1B,'1','2','3','4','5','6' ;* 00 Esc
X          DC.B      '7','8','9','0','-','=',$08,$09 ;* 08 BS,TAB
X          DC.B      'q','w','e','r','t','y','u','i' ;* 10
X          DC.B      'o','p',$5B,$5D,$0D,$00,'a','s' ;* 18 brackets,CR
X          DC.B      'd','f','g','h','j','k','l',';' ;* 20
X          DC.B      $27,'`',$00,'#','z','x','c','v' ;* 28 '
X          DC.B      'b','n','m',',','.','/',$00,$00 ;* 30
X          DC.B      $00,' ',$00,$00,$00,$00,$00,$00 ;* 38
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 40
X          DC.B      $00,$00,'-',$00,$00,$00,'+',$00 ;* 48
X          DC.B      $00,$00,$00,$7F,$00,$00,$00,$00 ;* 50 DEL
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 58
X          DC.B      $5C,$00,$00,'(',')','/','*','7' ;* 60 \
X          DC.B      '8','9','4','5','6','1','2','3' ;* 68
X          DC.B      '0','.',$0D,$00,$00,$00,$00,$00 ;* 70 Enter
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 78
Xshiftblk:
X          DC.B      $00,$1B,'!','@','#','$','%','^' ;* 00
X          DC.B      '&','*','(',')','_','+',$08,$09 ;* 08
X          DC.B      'Q','W','E','R','T','Y','U','I' ;* 10
X          DC.B      'O','P','{','}',$0D,$00,'A','S' ;* 18
X          DC.B      'D','F','G','H','J','K','L',':' ;* 20
X          DC.B      '"',$BA,$00,'~','Z','X','C','V' ;* 28
X          DC.B      'B','N','M','<','>','?',$00,$00 ;* 30
X          DC.B      $00,' ',$00,$00,$00,$00,$00,$00 ;* 38
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 40
```

```
X           DC.B      $00,$00,'-',$00,$00,$00,'+',$00 ;* 48
X           DC.B      $00,$00,$00,$7F,$00,$00,$00,$00 ;* 50
X           DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 58
X           DC.B      '|',$00,$00,'(',')','/','*','7' ;* 60
X           DC.B      '8','9','4','5','6','1','2','3' ;* 68
X           DC.B      '0','.',$0D,$00,$00,$00,$00,$00 ;* 70
X           DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 78
Xcapslock:
X           DC.B      $00,$1B,'1','2','3','4','5','6' ;* 00
X           DC.B      '7','8','9','0','-','=',$08,$09 ;* 08
X           DC.B      'Q','W','E','R','T','Y','U','I' ;* 10
X           DC.B      'O','P',$5B,$5D,$0D,$00,'A','S' ;* 18
X           DC.B      'D','F','G','H','J','K','L',';' ;* 20
X           DC.B      $27,'`',$00,'#','Z','X','C','V' ;* 28
X           DC.B      'B','N','M',',','.','/',$00,$00 ;* 30
X           DC.B      $00,' ',$00,$00,$00,$00,$00,$00 ;* 38
X           DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 40
X           DC.B      $00,$00,'-',$00,$00,$00,'+',$00 ;* 48
X           DC.B      $00,$00,$00,$7F,$00,$00,$00,$00 ;* 50
X           DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 58
X           DC.B      $5C,$00,$00,'(',')','/','*','7' ;* 60
X           DC.B      '8','9','4','5','6','1','2','3' ;* 68
X           DC.B      '0','.',$0D,$00,$00,$00,$00,$00 ;* 70
X           DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 78
XALTtabl:
X           DC.B      $00,$1B,$BB,$AB,$9C,$AC,$BF,$A9 ;* 00
X           DC.B      $BD,$F9,$F8,$B2,$7F,$F0,$00,$00 ;* 08
X           DC.B      $91,$C2,$89,$D5,$E7,$98,$81,$8B ;* 10
X           DC.B      $94,$E3,$C7,$F5,$00,$00,$84,$9E ;* 18
X           DC.B      $EB,$9F,$C4,$C6,$C0,$CC,$CD,$C8 ;* 20
X           DC.B      $CA,$CB,$00,$86,$B0,$B1,$87,$DE ;* 28
X           DC.B      $C3,$A4,$E6,$DB,$DC,$B3,$00,$00 ;* 30
X           DC.B      $00,$20,$00,$00,$00,$00,$00,$00 ;* 38
X           DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 40
X           DC.B      $00,$00,$D9,$00,$00,$00,$D8,$00 ;* 48
X           DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 50
X           DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 58
X           DC.B      $B4,$00,$00,$E0,$E1,$ED,$EF,$A1 ;* 60
X           DC.B      $8C,$8D,$A0,$83,$85,$82,$88,$8A ;* 68
X           DC.B      $E9,$E5,$00,$00,$00,$00,$00,$00 ;* 70
X           DC.B      $BB,$AB,$9C,$AC,$BF,$A9,$BD,$F9 ;* 78
X;* ALT & SHIFT
X           DC.B      $00,$1B,$AD,$B9,$DD,$9B,$F6,$AA ;* 00
X           DC.B      $BE,$FA,$A6,$A7,$F7,$F1,$00,$00 ;* 08
X           DC.B      $92,$D6,$EE,$D4,$D3,$9D,$9A,$EC ;* 10
X           DC.B      $99,$BC,$CF,$F4,$00,$00,$8E,$E4 ;* 18
X           DC.B      $C5,$CE,$E2,$C9,$C1,$D1,$D2,$D3 ;* 20
X           DC.B      $D4,$FF,$00,$8F,$B7,$B8,$80,$FB ;* 28
X           DC.B      $DF,$A5,$D0,$F3,$F2,$A8,$00,$00 ;* 30
X           DC.B      $00,$20,$00,$00,$00,$00,$00,$00 ;* 38
X           DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 40
X           DC.B      $00,$00,$DA,$00,$00,$00,$D7,$00 ;* 48
X           DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 50
```

```
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 58
X          DC.B      $B5,$00,$00,$AE,$AF,$E8,$B6,$A2 ;* 60
X          DC.B      $93,$95,$A3,$96,$97,$90,$FD,$FE ;* 68
X          DC.B      $EA,$FC,$00,$00,$00,$00,$00,$00 ;* 70
X          DC.B      $00,$00,$00,$00,$00,$00,$00,$00 ;* 78
X
X                                ;* data for saver
XX3:       DC.L      0
XW2:       DC.W      0
Xolacia:   DC.L      0
X
Xpglength:
X          DC.L      0
Xinstall:
X          MOVEA.L   4(A7),A5       ;* basepage
X          LEA       pglength(PC),A0
X          LEA       install(PC),A1
X          MOVE.L    A1,D0
X          SUB.L     A5,D0          ;* calculate length
X          MOVE.L    D0,(A0)
X          MOVE.L    D0,-(A7)       ;* length
X          MOVE.L    A5,-(A7)       ;* start
X          CLR.W     -(A7)
X          MOVE.W    #$4A,-(A7)
X          TRAP      #1             ;* SETBLOCK, reserve storage
X          LEA       12(A7),A7
X          LEA       messg1(PC),A0  ;* display message
X          BSR       outp
X          CLR.W     -(A7)          ;* read file DEFAULT.KBD
X          PEA       file1(PC)
X          MOVE.W    #$3D,-(A7)     ;* open
X          TRAP      #1
X          ADDQ.L    #8,A7
X          TST.W     D0
X          BMI.S     nodefault
X          PEA       normal(PC)     ;* table address for read
X          MOVE.L    #384,-(A7)     ;* length
X          MOVE.W    D0,-(A7)       ;* handle
X          MOVE.W    #$3F,-(A7)     ;* read
X          TRAP      #1
X          LEA       12(A7),A7
X          MOVE.W    handle(PC),-(A7)
X          MOVE.W    #$3E,-(A7)     ;* close
X          TRAP      #1
X          ADDQ.L    #4,A7
X          LEA       file1(PC),A0   ;* display file1
X          BSR       outp
Xnodefault:
X          LEA       messg2(PC),A0  ;* display message
X          BSR       outp
X          CLR.W     -(A7)          ;* read file ALT.KBD
X          PEA       file2(PC)
X          MOVE.W    #$3D,-(A7)     ;* open
```

```
X           TRAP      #1
X           ADDQ.L    #8,A7
X           TST.W     D0
X           BMI.S     noaltfile
X           LEA       handle(PC),A0  ;* file handle
X           MOVE.W    D0,(A0)
X           PEA       ALTtabl(PC)    ;* table address for read
X           MOVE.L    #256,-(A7)     ;* length
X           MOVE.W    D0,-(A7)       ;* handle
X           MOVE.W    #$3F,-(A7)     ;* read
X           TRAP      #1
X           LEA       12(A7),A7
X           MOVE.W    handle(PC),-(A7)
X           MOVE.W    #$3E,-(A7)     ;* close
X           TRAP      #1
X           ADDQ.L    #4,A7
X           LEA       file2(PC),A0   ;* display file2
X           BSR       outp
Xnoaltfile:
X           LEA       ALTtabl(PC),A0 ;* rearrange ALT-Codes
X           MOVE.L    $02(A0),$78(A0)
X           MOVE.L    $06(A0),$7C(A0)
X           MOVE.L    $0A(A0),$00(A0)
X           MOVE.L    $082(A0),$0F8(A0) ;* same for ALT SHIFT
X           MOVE.L    $086(A0),$0FC(A0)
X           MOVE.L    $08A(A0),$080(A0)
X           MOVE.W    #1,-(A7)
X           MOVE.W    #14,-(A7)
X           TRAP      #14            ;* get iorec(1)
X           ADDQ.L    #4,A7
X           LEA       IOrec(PC),A0
X           MOVE.L    D0,(A0)        ;* save pointer to kbd-iorec
X           MOVE.W    #34,-(A7)
X           TRAP      #14            ;* Kbdvbase()
X           ADDQ.L    #2,A7
X           MOVE.L    D0,A1          ;* A1 = addr(kbdvbase)
X           LEA       TOSvec,A0
X           MOVE.L    32(A1),(A0)    ;* save interrupt routine address
X           LEA       kbint,A0
X           MOVE.L    A0,32(A1)      ;* install new interrupt vectors
X           CLR.L     -(A7)
X           MOVE.W    #$20,-(A7)
X           TRAP      #1             ;* Super
X           ADDQ.L    #6,A7
X           LEA       TMRvec,A0
X           MOVE.L    $400,(A0)      ;* save etv_timer routine address
X           LEA       timerint,A0
X           MOVE.L    A0,$400        ;* install new interrupt vector
X           LEA       olacia,A0
X           MOVE.L    $118,(A0)      ;* save ACIA interrupt vector
X           LEA       acia,A0
X           MOVE.L    A0,$118        ;* install new interrupt vector
X           MOVE.L    $4F2,A0        ;* get system address
```

```
X          CMPI.B    #2,3(A0)       ;* check subversion number
X          BLT.S     oldsys         ;* TOS < 1.2: keep address
X          MOVE.W    26(A0),kbshadr+2  ;* copy kbshift variable address
Xoldsys:
X          MOVE.L    D0,-(A7)
X          MOVE.W    #$20,-(A7)
X          TRAP      #1
X          ADDQ.L    #6,A7
X          LEA       messg3(PC),A0  ;* newline
X          BSR.S     outp
X          CLR.W     -(A7)          ;* return code
X          MOVE.L    pglength(PC),-(A7) ;* program length
X          MOVE.W    #$31,-(A7)     ;* Ptermres
X          TRAP      #1             ;* reserve storage and end
Xoutp:     MOVE.L    A0,-(A7)
X          MOVE.W    #$09,-(A7)
X          TRAP      #1             ;* print line
X          ADDQ.L    #6,A7
X          RTS                      ;* never reached
Xhandle:   DC.W      0
Xmessg1:   DC.B      $0D,$0A,'KBEXT v. 4.10 installing keyboard ',$00
Xmessg2:   DC.B      $0D,$0A,'------------- and ALT table       ',$00
Xmessg3:   DC.B      $0D,$0A,$00
Xfile1:    DC.B      $5C,'AUTO',$5C
X          DC.B      'DEFAULT.KBD',$00
Xfile2:    DC.B      $5C,'AUTO',$5C
X          DC.B      'ALT.KBD',$00
X          END
SHAR_EOF
chmod 0600 KBEXT4.ASM || echo "restore of KBEXT4.ASM fails"
exit 0
```

```
From -4930537090373511849
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,b64fcfd0504e08db,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-11-30 22:45:55 PST
Path: utzoo!attcan!uunet!samsung!rex!ames!sun-barr!newstop!sun!grapevine!
koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i012: cal -- UNIXoid calendar program
Keywords: shar, C
Message-ID: <34829@grapevine.uucp>
Date: 25 Nov 89 19:58:34 GMT
Sender: news@grapevine.EBay
Reply-To: mlake@irscscm.ucb.com (Marshall Lake)
Lines: 363
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: mlake@irscscm.ucb.com (Marshall Lake)
Posting-number: Volume 3, Issue 12
```

```
Archive-name: cal

The enclosed source code is a UNIX-like calendar program.
It was developed using MWC.  The documentation is self-contained.

-------CUT-------
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".
# Contents:  ./cal.c
if test -f './cal.c' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'./cal.c'\"
else
echo shar: Extracting \"'./cal.c'\" \(12751 characters\)
sed "s/^X//" >'./cal.c' <<'END_OF_FILE'
/*
    cal:  a UNIX-like calendar program
          format:  cal [-nihf filename]
                   where n = the number of days to look ahead including
                             today, max 99, default 2
                         filename = file to look for instead of cal.txt

                    i  return information regarding CAL
                    h  return help with regard to CAL
                    f  use the following specified file as input to CAL
                       instead of cal.txt

    RETURNS:
    0 - CAL terminated successfully
    1 - bad calling format
    2 - "count of days" greater than 99 or longer than 2 positions
    3 - "count of days" = 0
    4 - cal.txt file not found
    5 - record in cal.txt greater than 1000 characters
    6 - information only requested (i and/or h options)

    Marshall Lake, TEAM Software, 4/26/88
    POB 7332
    Washington, DC  20044
    Tel 703-533-2132
    Fax 703-538-4598
    GEnie:  MLAKE
    CIS:  73717,3174
    uucp:   ...{media,teemc,tcsc3b2,ki4pv}!ka3ovk!irscscm!mlake
*/

#include <time.h>
#include <stdio.h>
#include <osbind.h>
#include <ctype.h>

long elapsed_time;
char today[26],
```

```
     filename[100] = "cal.txt",
     blanks[14] = "             ",
     alpha_months[37] = "JanFebMarAprMayJunJulAugSepOctNovDec",
     full_months[109] = "january  february march    april    may      june\
     july     august   septemberoctober  november december ";
int  orig_day, day,
     orig_month, month;

main (argc, argv)
int argc;
char *argv[];
{
    int  x,
         y,
         l,
         set_i, set_f, set_h, set_n,
         count_of_days,
         loop;
    char cal_rec[1001],
         mat1[5],            /* mm/dd */
         mat11[4],           /* m/dd */
         mat12[4],           /* mm/d */
         mat13[3],           /* m/d */
         mat2[5],            /* mm-dd */
         mat21[4],           /* m-dd */
         mat22[4],           /* mm-d */
         mat23[3],           /* m-d */
         mat3[6],            /* aaa dd */
         mat31[5],           /* aaa d */
         mat4[6],            /* dd aaa */
         mat41[5],           /* d aaa */
         mat5[13],           /* aaaaaaaaa dd */
         mat51[12],          /* aaaaaaaaa d */
         mat6[13],           /* dd aaaaaaaaa */
         mat61[12];          /* d aaaaaaaaa */
    FILE *stream;

    if (argc > 3)
        bad_format ();

    set_i = set_h = set_f = set_n = 0;
    if (argc > 1) {
        if (strlen (argv[1]) > 6 || argv[1][0] != '-')
            bad_format ();
        for (x = 1; x < strlen (argv[1]); x++)
            if (argv[1][x] >= '0' && argv[1][x] <= '9') {
                if (set_n == 1)
                    bad_format ();
                set_n = 1;
                if (argv[1][x + 1] >= '0' && argv[1][x + 1] <= '9') {
                    if (argv[1][x + 2] >= '0' && argv[1][x + 2] <= '9') {
                        fprintf (stderr, " max count 2 digits\n\n");
                        exit (2);
```

```
                }
                count_of_days = (argv[1][x] - '0') * 10 +
                                (argv[1][x + 1] - '0');
                x++;
            }
            else
                count_of_days = argv[1][x] - '0';
        }
        else
            if (argv[1][x] == 'f') {
                if (set_f == 1 || argc != 3)
                    bad_format ();
                set_f = 1;
            }
            else
                if (argv[1][x] == 'i') {
                    if (set_i == 1)
                        bad_format ();
                    set_i = 1;
                }
                else
                    if (argv[1][x] == 'h') {
                        if (set_h == 1)
                            bad_format ();
                        set_h = 1;
                    }
                    else
                        bad_format ();
}
if (set_n == 0)
    count_of_days = 2;
if (set_f == 1)
    strcpy (&filename[0], argv[2]);
if (set_i == 1) {
    printf ("\n\n CAL: a UNIX-like calendar program\n");
    printf (" written by M. Lake\n");
    printf (" v1.3  7/28/88\n\n");
}
if (set_h == 1)
    print_info ();

if (count_of_days == 0)
    exit (3);
if (set_i == 1 || set_h == 1 && set_n == 0 && set_f == 0)
    exit (6);

time (&elapsed_time);
strncpy (&today[0], ctime (&elapsed_time), 26);
orig_day = (today[8] - '0') * 10 + (today[9] - '0');
for (x = 0; x < 34; x += 3)
    if (strncmp (&today[4], &alpha_months[x], 3) == 0)
        orig_month = x / 3 + 1;
```

```
    if ((stream = fopen (filename, "rb")) == NULL) {
        fprintf (stderr,
                 "%s file does not exist where it is expected!\n\n", filename);
        exit (4);
    }

    mat1[2] = mat11[1] = mat12[2] = mat13[1] = '/';
    mat2[2] = mat21[1] = mat22[2] = mat23[1] = '-';
    mat3[3] = mat4[2] = mat31[3] = mat41[1] = ' ';

    cal_rec[1000] = '\0';

    while (1) {
next_record:
        month = orig_month;
        day = orig_day;
        if ((fgets (&cal_rec[0], 1001, stream)) == 0) {
            fclose (stream);
            exit (0);
        }
        if (strlen (&cal_rec[0]) == 1000) {
            fprintf (stderr, "\n\nA record in %s is too big!!\n", filename);
            fprintf (stderr, "1000 characters is the largest allowed.\n");
            fclose (stream);
            exit (5);
        }

        l = strlen (&cal_rec[0]) - 1;
        for (loop = 0; loop < count_of_days; loop++) {
            strncpy (&mat5[0], &blanks[0], 13);
            strncpy (&mat6[0], &blanks[0], 13);
            strncpy (&mat51[0], &blanks[0], 12);
            strncpy (&mat61[0], &blanks[0], 12);
            mat1[0] = mat2[0] = mat12[0] = mat22[0] = month / 10 + '0';
            mat1[1] = mat2[1] = mat11[0] = mat21[0] = mat12[1] =
              mat22[1] = mat13[0] = mat23[0] = month % 10 + '0';
            mat1[3] = mat2[3] = mat3[4] = mat4[0] = mat6[0] = mat11[2] =
              mat21[2] = day / 10 + '0';
            mat1[4] = mat2[4] = mat3[5] = mat4[1] = mat6[1] = mat11[3] =
              mat21[3] = mat12[3] = mat22[3] = mat13[2] = mat23[2] =
              mat31[4] = mat41[0] = mat61[0] = day % 10 + '0';
            for (x = (month - 1) * 3, y = 0; y < 3; x++, y++)
                mat3[y] = mat4[y + 3] = mat31[y] = mat41[y + 2] =
                  alpha_months[x];
            for (x = (month - 1) * 9, y = 0; y < 9; x++, y++)
                mat5[y] = mat6[y + 3] = mat51[y] = mat61[y + 2] =
                   full_months[x];
            for (x = 3; x < 12; x++)
                if (mat6[x] == ' ') {
                    mat6[x] = mat5[x] = mat61[x - 1] = mat51[x - 1] = '\0';
                    break;
                }
            mat5[x - 2] = day / 10 + '0';
```

```
        mat5[x - 1] = mat51[x - 2] = day % 10 + '0';

        for (x = 0; (x + 2) < l; x++) {
            if (strncmp (&cal_rec[x], &mat1[0], 5) == 0 ||
                strncmp (&cal_rec[x], &mat2[0], 5) == 0 ||
                strnicmp (&cal_rec[x], &mat3[0], 6) == 0 ||
                strnicmp (&cal_rec[x], &mat5[0], strlen (&mat5[0])) == 0 ||
                strnicmp (&cal_rec[x], &mat6[0], strlen (&mat6[0])) == 0 ||
                strnicmp (&cal_rec[x], &mat4[0], 6) == 0) {
                printf ("%s", cal_rec);
                goto next_record;
            }
            if (month < 10 && day < 10)
                if ((strncmp (&cal_rec[x], &mat13[0], 3) == 0 ||
                     strncmp (&cal_rec[x], &mat23[0], 3) == 0) &&
                    (cal_rec[x - 1] < '0' || cal_rec[x - 1] > '9') &&
                    (cal_rec[x + 3] < '0' || cal_rec[x + 3] > '9')) {
                    printf ("%s", cal_rec);
                    goto next_record;
                }
            if (month < 10)
                if ((strncmp (&cal_rec[x], &mat11[0], 4) == 0 ||
                     strncmp (&cal_rec[x], &mat21[0], 4) == 0) &&
                    (cal_rec[x - 1] < '0' || cal_rec[x - 1] > '9')) {
                    printf ("%s", cal_rec);
                    goto next_record;
                }
            if (day < 10) {
                if (((strncmp (&cal_rec[x], &mat12[0], 4) == 0 ||
                      strncmp (&cal_rec[x], &mat22[0], 4) == 0) &&
                      (cal_rec[x + 4] < '0' || cal_rec[x + 4] > '9')) ||
                      (strnicmp (&cal_rec[x], &mat31[0], 5) == 0 &&
                      (cal_rec[x + 5] < '0' || cal_rec[x + 5] > '9')) ||
                      (strnicmp (&cal_rec[x], &mat41[0], 5) == 0 &&
                      (cal_rec[x - 1] < '0' || cal_rec[x - 1] > '9'))) {
                    printf ("%s", cal_rec);
                    goto next_record;
                 }
                 if ((strnicmp (&cal_rec[x], &mat51[0],
                         strlen (&mat51[0])) == 0 &&
                      (cal_rec[x + strlen (&mat51[0])] < '0' ||
                         cal_rec[x + strlen (&mat51[0])] > '9')) ||
                      (strnicmp (&cal_rec[x], &mat61[0],
                         strlen (&mat61[0])) == 0 &&
                      (cal_rec[x - 1] < '0' || cal_rec[x - 1] > '9'))) {
                    printf ("%s", cal_rec);
                    goto next_record;
                }
            }
        }
        add_a_day ();
    }
}
```

```
}

add_a_day () {
    day++;
    if (day == 29 && month == 2) {
        day = 1;
        month++;
    }
    if (day == 31 && (month == 9 || month == 4 ||
                      month == 6 || month == 11)) {
        day = 1;
        month++;
    }
    if (day == 32)
        if (month == 12)
            day = month = 1;
        else {
            day = 1;
            month++;
        }
}

bad_format () {
    fprintf (stderr, "\n format: 'cal [-nihf filename]'\n");
    fprintf (stderr, " where n = count of days to check\n");
    fprintf (stderr, "\n enter 'cal -h' and press <RETURN> for help\n");
    exit (1);
}

/* check for a match with no regard to case */

strnicmp (a, b, length)
char *a, *b;
int length;
{
    char a_fld[100], b_fld[100];
    int x;

    strncpy (&a_fld[0], a, length);
    strncpy (&b_fld[0], b, length);

    for (x = 0; x < length; x++) {
        if (isupper (a_fld[x]))
            a_fld[x] = tolower (a_fld[x]);
        if (isupper (b_fld[x]))
            b_fld[x] = tolower (b_fld[x]);
    }

    return (strncmp (&a_fld[0], &b_fld[0], length));
}

print_info () {
    printf ("\n\ncal:  a UNIX-like calendar program\n\n");
```

```
    printf ("format:  cal [-nihf filename]\n");
    printf ("  where n = the number of days to look ahead including\n");
    printf ("        today, max 99, default 2\n");
    printf ("    filename = file to look for instead of cal.txt\n\n");
    printf ("   i  return information regarding CAL\n");
    printf ("   h  return help with regard to CAL\n");
    printf ("   f  use the following specified file as input to CAL\n");
    printf ("      instead of cal.txt\n\n");
    printf ("CAL will return records from a 'cal.txt' file residing in\n");
    printf ("the same directory as where 'cal.prg' is called from.\n");
    printf ("CAL will seek out and return records containing dates that\n");
    printf ("are being requested.  A record is a string of characters\n");
    printf ("ending with a carriage return.\n\n");
    printf (" ** PRESS ANY KEY **\n");
    Crawcin ();
    printf ("\n\nThe format of the date may be any of the following:\n");
    printf ("mm/dd, m/dd, mm/d, m/d\n");
    printf ("mm-dd, m-dd, mm-d, m-d\n");
    printf ("aaa dd, aaa d\n");
    printf ("dd aaa, d aaa\n");
    printf ("aaaaaaaaa dd, aaaaaaaaa d\n");
    printf ("dd aaaaaaaaa, d aaaaaaaaa\n\n");
    printf ("where mm or m = numeric month (1 - 12), dd or d = numeric\n");
    printf ("day (1 - 31), aaa = the first three characters of the\n");
    printf ("alpha month (jan - dec), aaaaaaaaa = the entire alpha\n");
    printf ("month (january - december) and is variable length.\n\n");
    printf ("CAL is NOT case sensitive with regard to alpha months.\n");
    printf ("'cal.txt' may be any size up to the physical limitations\n");
    printf ("of your machine.\n");
    printf ("The largest record allowed is 1000 characters.\n\n");
    printf (" ** PRESS ANY KEY **\n");
    Crawcin ();
    printf ("\n\nRESERVATIONS:\n");
    printf ("CAL will not recognize years.\n");
    printf ("The internal date in your machine must be correct to\n");
    printf ("enable CAL to return the correct information.\n\n");
}
END_OF_FILE
if test 12751 -ne `wc -c <'./cal.c'`; then
    echo shar: \"'./cal.c'\" unpacked with wrong size!
fi
# end of './cal.c'
fi


From 9124068172435697339
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,6dd50f1802d1b2c5,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-12-09 11:56:51 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!clyde.concordia.ca!uunet!cs.utexas.edu!
sun-barr!newstop!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
```

Newsgroups: comp.sources.atari.st
Subject: v03i013: dhrystone -- Dhrystone benchmark program
Keywords: shar, C
Message-ID: <34934@grapevine.uucp>
Date: 9 Dec 89 19:56:51 GMT
Sender: news@grapevine.uucp
Reply-To: rosenkra@hall.cray.com (Bill Rosenkranz)
Lines: 1348
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: rosenkra@hall.cray.com (Bill Rosenkranz)
Posting-number: Volume 3, Issue 13
Archive-name: dhrystone

this is a port of the dhrystone bm, v2.0, i found on the MIPS BBS. when
run on my mega4, compiled with alcyon 4.14, it ran at about 719. see
the file OUT.ST for the results of that run.

-bill
net: rosenkra@hall.cray.com
CIS: 71460,17

--------cut here--------------cut here----------------cut here---------------
This is a shell archive. Remove anything before this line,
then unpack it by saving it in a file and typing "sh file".
#
Wrapped by hall!rosenkra on Tue Dec 5 23:34:28 CST 1989
Contents: MANIFEST Makefile README OUT.ST config.h dhry_glo.h dhry_p1.c
dhry_p2.c

echo x - MANIFEST
sed 's/^@//' > "MANIFEST" <<'@//E*O*F MANIFEST//'
total 66
-rw-r--r-- 1 rosenkra   550 Dec 5 23:30 MANIFEST
-rw-r--r-- 1 rosenkra   935 Dec 5 23:22 Makefile
-rw-r--r-- 1 rosenkra   1773 Dec 5 23:22 OUT.ST
-rw-r--r-- 1 rosenkra   16178 Dec 5 23:22 README
-rw-r--r-- 1 rosenkra   235 Dec 5 23:30 README.ST
-rw-r--r-- 1 rosenkra   712 Dec 5 23:22 config.h
-rw-r--r-- 1 rosenkra   2189 Dec 5 23:22 dhry_glo.h
-rw-r--r-- 1 rosenkra   10993 Dec 5 23:22 dhry_p1.c
-rw-r--r-- 1 rosenkra   4456 Dec 5 23:22 dhry_p2.c
@//E*O*F MANIFEST//
chmod u=rw,g=r,o=r MANIFEST

echo x - Makefile
sed 's/^@//' > "Makefile" <<'@//E*O*F Makefile//'
makefile for Dhrystone v2.0 for atariST (alcyon)
#
first edit config.h for your system/compiler...
#
UUENCODE = c:\bin\uue.ttp

```
TARGET        = dhry.ttp
ARCFILE       = dhry20st.arc
SHARFILE  = dhry20st.shr
UUFILE        = dhry.uue
SRCS      = config.h dhry_glo.h dhry_p1.c dhry_p2.c
OTHERS        = readme makefile
DISTFILES= $(OTHERS) $(SRCS)
OBJS      = dhry_p1.o dhry_p2.o
LIBS      = -llibc
LDFLAGS       = -s -new
CFLAGS        = -new


# main target...
#
$(TARGET):    $(OBJS)
          $(CC) $(LDFLAGS) -o $(TARGET) $(OBJS) $(LIBS)


# to run it...
#
run:      $(TARGET)
          $(TARGET) 30000

clean:
          rm $(OBJS) errs

clobber:  clean
          rm $(TARGET)

arc:
          $(ARC) a $(ARCFILE) $(DISTFILES)
          @echo done making $(ARCFILE)

uue:
          $(UUENCODE) $(TARGET)
          @echo done making $(UUFILE)

shar:
          $(SHAR) $(DISTFILES) >$(SHARFILE)
          @echo done making $(SHARFILE)

# dependencies...
#
dhry_glo.h:   config.h
          $(TOUCH) dhry_glo.h

dhry_p1.o:    dhry_glo.h dhry_p1.c
dhry_p2.o:    dhry_glo.h dhry_p2.c

@//E*O*F Makefile//
chmod u=rw,g=r,o=r Makefile
```

```
echo x - README
sed 's/^@//' > "README" <<'@//E*O*F README//'
/* NOTE: edit config.h first for your system!!! */

/*
 *************************************************************************
 *
 *                  "DHRYSTONE" Benchmark Program
 *                  -----------------------------
 *
 *  Version:    C, Version 2.0
 *
 *  Date:       March 3, 1988
 *
 *  Author:     Reinhold P. Weicker
 *                      Siemens AG, E STE 35
 *                      Postfach 3240
 *                      8520 Erlangen
 *                      Germany (West)
 *                              Phone:  [xxx-49]-9131-7-20330
 *                                      (8-17 Central European Time)
 *                              Usenet: ..!mcvax!unido!estevax!weicker
 *
 *              Original Version (in Ada) published in
 *              "Communications of the ACM" vol. 27., no. 10 (Oct. 1984),
 *              pp. 1013 - 1030, together with the statistics
 *              on which the distribution of statements etc. is based.
 *
 *              In this C version, the following C library functions are used:
 *              - strcpy, strcmp (inside the measurement loop)
 *              - printf, scanf (outside the measurement loop)
 *              In addition, UNIX system calls "times ()" or "time ()"
 *              are used for execution time measurement. For measurements
 *              on other systems, these calls have to be changed.
 *
 *  Collection of Results:
 *              Reinhold Weicker (address see above) and
 *
 *              Rick Richardson
 *              PC Research. Inc.
 *              94 Apple Orchard Drive
 *              Tinton Falls, NJ 07724
 *                      Phone:  (201) 834-1378 (9-17 EST)
 *                      Usenet: ...!seismo!uunet!pcrat!rick
 *
 *      Please send results to Reinhold Weicker and/or Rick Richardson.
 *      Complete information should be given on hardware and software used.
 *      Hardware information includes: Machine type, CPU, type and size
 *      of caches; for microprocessors: clock frequency, memory speed
 *      (number of wait states).
 *      Software information includes: Compiler (and runtime library)
 *      manufacturer and version, compilation switches, OS version.
 *      The Operating System version may give an indication about the
```

```
 *      compiler; Dhrystone itself performs no OS calls in the measurement loop.
 *
 *      The complete output generated by the program should be mailed
 *      such that at least some checks for correctness can be made.
 */
/*
 ****************************************************************************
 *
 *  History:    This version C/2.0 has been made for two reasons:
 *
 *              1) There is an obvious need for a common C version of
 *              Dhrystone, since C is at present the most popular system
 *              programming language for the class of processors
 *              (microcomputers, minicomputers) where Dhrystone is used most.
 *              There should be, as far as possible, only one C version of
 *              Dhrystone such that results can be compared without
 *              restrictions. In the past, the C versions distributed
 *              by Rick Richardson (Version 1.1) and by Reinhold Weicker
 *              had small (though not significant) differences.
 *
 *              2) As far as it is possible without changes to the Dhrystone
 *              statistics, optimizing compilers should be prevented from
 *              removing significant statements.
 *
 *              This C version has been developed in cooperation with
 *              Rick Richardson (Tinton Falls, NJ), it incorporates many
 *              ideas from the "Version 1.1" distributed previously by
 *              him over the UNIX network Usenet.
 *              I also thank Chaim Benedelac (National Semiconductor),
 *              David Ditzel (SUN), Earl Killian and John Mashey (MIPS),
 *              Alan Smith and Rafael Saavedra-Barrera (UC at Berkeley)
 *              for their help with comments on earlier versions of the
 *              benchmark.
 *
 *  Changes:    In the initialization part, this version follows mostly
 *              Rick Richardson's version distributed via Usenet, not the
 *              version distributed earlier via floppy disk by Reinhold Weicker.
 *              As a concession to older compilers, names have been made
 *              unique within the first 8 characters.
 *              Inside the measurement loop, this version follows the
 *              version previously distributed by Reinhold Weicker.
 *
 *              At several places in the benchmark, code has been added,
 *              but within the measurement loop only in branches that
 *              are not executed. The intention is that optimizing compilers
 *              should be prevented from moving code out of the measurement
 *              loop, or from removing code altogether. Since the statements
 *              that are executed within the measurement loop have NOT been
 *              changed, the numbers defining the "Dhrystone distribution"
 *              (distribution of statements, operand types and locality)
 *              still hold. Except for sophisticated optimizing compilers,
 *              execution times for this version should be the same as
 *              for previous versions.
```

```
 *
 *              Since it has proven difficult to subtract the time for the
 *              measurement loop overhead in a correct way, the loop check
 *              has been made a part of the benchmark. This does have
 *              an impact - though a very minor one - on the distribution
 *              statistics which have been updated for this version.
 *
 *              All changes within the measurement loop are described
 *              and discussed in the companion paper "Rationale for
 *              Dhrystone version 2".
 *
 *              Because of the self-imposed limitation that the order and
 *              distribution of the executed statements should not be
 *              changed, there are still cases where optimizing compilers
 *              may not generate code for some statements. To a certain
 *              degree, this is unavoidable for small synthetic benchmarks.
 *              Users of the benchmark are advised to check code listings
 *              whether code is generated for all statements of Dhrystone.
 */
/*
 ************************************************************************
 *
 * Defines:     The following "Defines" are possible:
 *              -DREG=register          (default: Not defined)
 *                      As an approximation to what an average C programmer
 *                      might do, the "register" storage class is applied
 *                      (if enabled by -DREG=register)
 *                      - for local variables, if they are used (dynamically)
 *                        five or more times
 *                      - for parameters if they are used (dynamically)
 *                        six or more times
 *                      Note that an optimal "register" strategy is
 *                      compiler-dependent, and that "register" declarations
 *                      do not necessarily lead to faster execution.
 *              -DNOSTRUCTASSIGN        (default: Not defined)
 *                      Define if the C compiler does not support
 *                      assignment of structures.
 *              -DNOENUMS               (default: Not defined)
 *                      Define if the C compiler does not support
 *                      enumeration types.
 *              -DTIMES                 (default)
 *              -DTIME
 *                      The "times" function of UNIX (returning process times)
 *                      or the "time" function (returning wallclock time)
 *                      is used for measurement.
 *                      For single user machines, "time ()" is adequate. For
 *                      multi-user machines where you cannot get single-user
 *                      access, use the "times ()" function. If you have
 *                      neither, use a stopwatch in the dead of night.
 *                      "printf"s are provided marking the points "Start Timer"
 *                      and "Stop Timer". DO NOT use the UNIX "time(1)"
 *                      command, as this will measure the total time to
 *                      run this program, which will (erroneously) include
```

```
 *                      the time to allocate storage (malloc) and to perform
 *                      the initialization.
 *              -DHZ=nnn                (default: 60)
 *                      The function "times" returns process times in
 *                      1/HZ seconds, with HZ = 60 for most systems.
 *                      CHECK YOUR SYSTEM DESCRIPTION BEFORE YOU JUST APPLY
 *                      THE DEFAULT VALUE.
 */
/*
 ************************************************************************
 *
 *  Compilation model and measurement (IMPORTANT):
 *
 *  This C version of Dhrystone consists of three files:
 *  - dhry_global.h (this file, containing global definitions and comments)
 *  - dhry_pack_1.c (containing the code corresponding to Ada package Pack_1)
 *  - dhry_pack_2.c (containing the code corresponding to Ada package Pack_2)
 *
 *  The following "ground rules" apply for measurements:
 *  - Separate compilation
 *  - No procedure merging
 *  - Otherwise, compiler optimizations are allowed but should be indicated
 *  - Default results are those without register declarations
 *  See the companion paper "Rationale for Dhrystone Version 2" for a more
 *  detailed discussion of these ground rules.
 *
 *  For 16-Bit processors (e.g. 80186, 80286), times for all compilation
 *  models ("small", "medium", "large" etc.) should be given if possible,
 *  together with a definition of these models for the compiler system used.
 */
/*
 ************************************************************************
 *
 *  Dhrystone (C version) statistics:
 *
 *  [Comment from the first distribution, updated for version 2.
 *   Note that because of language differences, the numbers are slightly
 *   different from the Ada version.]
 *
 *  The following program contains statements of a high level programming
 *  language (here: C) in a distribution considered representative:
 *
 *    assignments                   52 (51.0 %)
 *    control statements            33 (32.4 %)
 *    procedure, function calls     17 (16.7 %)
 *
 *  103 statements are dynamically executed. The program is balanced with
 *  respect to the three aspects:
 *
 *    - statement type
 *    - operand type
 *    - operand locality
 *         operand global, local, parameter, or constant.
```

```
*
*  The combination of these three aspects is balanced only approximately.
*
*  1. Statement Type:
*  -----------------              number
*
*     V1 = V2                      9
*       (incl. V1 = F(..)
*     V = Constant                12
*     Assignment,                  7
*       with array element
*     Assignment,                  6
*       with record component
*                                 --
*                                 34       34
*
*     X = Y +|-|"&&"|"|" Z         5
*     X = Y +|-|"==" Constant      6
*     X = X +|- 1                  3
*     X = Y *|/ Z                  2
*     X = Expression,              1
*           two operators
*     X = Expression,              1
*           three operators
*                                 --
*                                 18       18
*
*     if ....                     14
*       with "else"      7
*       without "else"   7
*           executed        3
*           not executed    4
*     for ...                      7  |  counted every time
*     while ...                    4  |  the loop condition
*     do ... while                 1  |  is evaluated
*     switch ...                   1
*     break                        1
*     declaration with             1
*       initialization
*                                 --
*                                 34       34
*
*     P (...)  procedure call     11
*       user procedure      10
*       library procedure    1
*     X = F (...)
*             function  call       6
*       user function         5
*       library function      1
*                                 --
*                                 17       17
*                                         ---
*                                        103
```

```
*
*    The average number of parameters in procedure or function calls
*    is 1.82 (not counting the function values as implicit parameters).
*
*
*  2. Operators
*  ------------
*                           number    approximate
*                                     percentage
*
*    Arithmetic              32          50.8
*
*       +                      21          33.3
*       -                       7          11.1
*       *                       3           4.8
*       / (int div)             1           1.6
*
*    Comparison              27           42.8
*
*       ==                      9           14.3
*       /=                      4           6.3
*       >                       1           1.6
*       <                       3           4.8
*       >=                      1           1.6
*       <=                      9           14.3
*
*    Logic                    4            6.3
*
*       && (AND-THEN)           1           1.6
*       |  (OR)                1           1.6
*       !  (NOT)               2           3.2
*
*                           --          -----
*                           63          100.1
*
*
*  3. Operand Type (counted once per operand reference):
*  ---------------
*                           number    approximate
*                                     percentage
*
*     Integer               175        72.3 %
*     Character              45        18.6 %
*     Pointer                12         5.0 %
*     String30                6         2.5 %
*     Array                   2         0.8 %
*     Record                  2         0.8 %
*                           ---       -------
*                           242       100.0 %
*
*  When there is an access path leading to the final operand (e.g. a record
*  component), only the final data type on the access path is counted.
*
```

```
 *
 *  4. Operand Locality:
 *  --------------------
 *                                 number    approximate
 *                                           percentage
 *
 *     local variable               114        47.1 %
 *     global variable               22         9.1 %
 *     parameter                     45        18.6 %
 *     value                         23         9.5 %
 *     reference                     22         9.1 %
 *     function result                6         2.5 %
 *     constant                      55        22.7 %
 *                                  ---       -------
 *                                  242       100.0 %
 *
 *
 *  The program does not compute anything meaningful, but it is syntactically
 *  and semantically correct. All variables have a value assigned to them
 *  before they are used as a source operand.
 *
 *  There has been no explicit effort to account for the effects of a
 *  cache, or to balance the use of long or short displacements for code or
 *  data.
 *
 ****************************************************************************
 */

@//E*O*F README//
chmod u=rw,g=r,o=r README

echo x - OUT.ST
sed 's/^@//' > "OUT.ST" <<'@//E*O*F OUT.ST//'
    dhry.ttp 30000

Dhrystone Benchmark, Version 2.0 (Language: C)

System:   Atari MegsST 4 (TOS, 8 Mhz)
Compiler: DRI/Alcyon 4.14

Program compiled with 'register' attribute


Execution starts, 30000 runs through Dhrystone
Execution ends

Final values of the variables used in the benchmark:

Int_Glob:            5
        should be:   5
Bool_Glob:           1
        should be:   1
Ch_1_Glob:           A
```

```
        should be:   A
Ch_2_Glob:           B
        should be:   B
Ar_1_Glob[8]:        7
        should be:   7
Ar_2_Glob[8][7]:     30010
        should be:   Number_Of_Runs + 10
Ptr_Glob->
  Ptr_Comp:          22250
        should be:   (implementation-dependent)
  Discr:             0
        should be:   0
  Enum_Comp:         2
        should be:   2
  Int_Comp:          17
        should be:   17
  Str_Comp:          DHRYSTONE PROGRAM, SOME STRING
        should be:   DHRYSTONE PROGRAM, SOME STRING
Next_Ptr_Glob->
  Ptr_Comp:          22250
        should be:   (implementation-dependent), same as above
  Discr:             0
        should be:   0
  Enum_Comp:         1
        should be:   1
  Int_Comp:          18
        should be:   18
  Str_Comp:          DHRYSTONE PROGRAM, SOME STRING
        should be:   DHRYSTONE PROGRAM, SOME STRING
Int_1_Loc:           5
        should be:   5
Int_2_Loc:           13
        should be:   13
Int_3_Loc:           7
        should be:   7
Enum_Loc:            1
        should be:   1
Str_1_Loc:           DHRYSTONE PROGRAM, 1'ST STRING
        should be:   DHRYSTONE PROGRAM, 1'ST STRING
Str_2_Loc:           DHRYSTONE PROGRAM, 2'ND STRING
        should be:   DHRYSTONE PROGRAM, 2'ND STRING

Microseconds for one run through Dhrystone: 1449.9
Dhrystones per Second:                       719.2

@//E*O*F OUT.ST//
chmod u=rw,g=r,o=r OUT.ST

echo x - config.h
sed 's/^@//' > "config.h" <<'@//E*O*F config.h//'
/*
 *   config.h - configuration for Dhrystone 2.0
 *
```

```
 *   choices:
 *        -DREG=register
 *              -DNOSTRUCTASSIGN
 *              -DNOENUMS
 *              -DTIMES
 *              -DTIME
 *              -DHZ=nnn
 */

/*
 *   atariST (TOS, dri/alcyon 4.14)
 *
 *   int is 16 bits here. so max times thru is < 32k times.
 *
 */
#define ALCYON
#define SYSTYPE         "Atari MegsST 4 (TOS 1.2, 8 Mhz 68000)"
#define COMPTYPE   "DRI/Alcyon 4.14 (16-bit ints)"
#define REG        register
#define TIME
#define NOENUM

/*
 *   BSD
 *
#define SYSTYPE         "BSD Unix"
#define COMPTYPE   "pcc"
#define REG        register
#define TIME
 */

/*
 *   SYSV
 *
#define SYSTYPE         "System V.3 Unix"
#define COMPTYPE   "pcc"
#define REG        register
#define TIME
 */

@//E*O*F config.h//
chmod u=rw,g=r,o=r config.h

echo x - dhry_glo.h
sed 's/^@//' > "dhry_glo.h" <<'@//E*O*F dhry_glo.h//'
#include "config.h"

/*
 ************************************************************************
 *
 *                   "DHRYSTONE" Benchmark Program
 *                   -----------------------------
 *
```

```
 *  Version:    C, Version 2.0
 *
 *  File:       dhry_glo.h (part 1 of 3)
 *
 *  Date:       March 3, 1988
 *
 *  Author:     Reinhold P. Weicker
 *
 ************************************************************************
 */

/*
 *   Compiler and system dependent definitions:
 */
#ifndef TIME
#ifndef TIMES          /* Use "times" function for measurement */
#define TIMES          /* unless explicitly defined otherwise  */
#endif
#endif

#ifndef HZ
#define HZ         60   /* Use HZ = 60 for "times" function     */
#endif                  /* unless explicitly defined otherwise  */

#ifdef TIMES
#include <sys/types.h>
#include <sys/times.h>      /* for "times" */
#endif

#define Usec_Per_Second 1000000.0/* UNIX C returns process times in seconds/HZ */

#ifdef  NOSTRUCTASSIGN
#define str_asgn(d,s)   memcpy(&(d), &(s), sizeof(d))
#else
#define str_asgn(d,s)   d = s
#endif

/*
 *   for boolean and enumeration types in Ada, Pascal
 */
#ifdef  NOENUM
#define Ident_1         0
#define Ident_2         1
#define Ident_3         2
#define Ident_4         3
#define Ident_5         4
#define Enumeration     int
#else
typedef enum {Ident_1, Ident_2, Ident_3, Ident_4, Ident_5} Enumeration;
#endif


/*
```

```
 *   General definitions:
 */
#include <stdio.h>          /* for strcpy, strcmp */

#define Null       0        /* Value of a Null pointer */
#define true       1
#define false      0
#define One_Thirty int
#define One_Fifty  int
#define Capital_Letter  char
#define Boolean         int
#define Str_30          char
#define Array      int
#define Rec_Type   struct record
#define Rec_Pointer    struct recort

struct record
{
     struct record            *Ptr_Comp;
     Enumeration              Discr;

     union
     {
          struct
          {
               Enumeration   Enum_Comp;
               int       Int_Comp;
               char      Str_Comp[31];

          }              var_1;
          struct
          {
               Enumeration   E_Comp_2;
               char      Str_2_Comp[31];

          }              var_2;
          struct
          {
               char      Ch_1_Comp;
               char      Ch_2_Comp;

          }              var_3;

     }                   variant;
};

@//E*O*F dhry_glo.h//
chmod u=rw,g=r,o=r dhry_glo.h

echo x - dhry_p1.c
sed 's/^@//' > "dhry_p1.c" <<'@//E*O*F dhry_p1.c//'
/*
 ****************************************************************************
```

```
 *
 *                   "DHRYSTONE" Benchmark Program
 *                   -----------------------------
 *
 *  Version:    C, Version 2.0
 *
 *  File:       dhry_pack_1.c (part 2 of 3)
 *
 *  Date:       March 3, 1988
 *
 *  Author:     Reinhold P. Weicker
 *
 ***************************************************************************
 */

#include "dhry_glo.h"

/*
 *   Global Variables:
 */
Rec_Pointer    *Ptr_Glob,
           *Next_Ptr_Glob;
int      Int_Glob;
Boolean  Bool_Glob;
char     Ch_1_Glob,
         Ch_2_Glob;
int      Ar_1_Glob[50];
int      Ar_2_Glob[50][50];
char     System_Type[80];
char     Compiler_Type[80];

/*
 *   forward declaration necessary since Enumeration may not simply be int
 */
extern char    *malloc ();
Enumeration   Func_1 ();

#ifndef REG
Boolean  Reg = false;       /* REG becomes defined as empty */
#define REG                 /* i.e. no register variables   */
#else
Boolean  Reg = true;
#endif

/*
 *   variables for time measurement:
 */
#ifdef TIMES
struct tms      time_info;
extern int      times ();       /* see library function "times" */
#endif
#ifdef TIME
extern long     time ();        /* see library function "time"  */
```

```
#endif

#define Too_Small_Time  2        /* Measurements should last at least 2
                           seconds */

long     Begin_Time,
            End_Time,
         User_Time;
float    Microseconds,
            Dhrystones_Per_Second;



main (argc, argv)
int     argc;
char   *argv[];
{

/*
 *   main program, corresponds to procedures Main and Proc_0 in the
 *   Ada version
 */

    One_Fifty Int_1_Loc;
    REG One_Fifty      Int_2_Loc;
    One_Fifty     Int_3_Loc;
    REG char        Ch_Index;
    Enumeration   Enum_Loc;
    Str_30        Str_1_Loc[31];
    Str_30        Str_2_Loc[31];
    REG int   Run_Index;
    REG int   Number_Of_Runs;
    long      dum = 0L;


    /*
     *   Initializations
     */
    Next_Ptr_Glob = (Rec_Pointer *) malloc (sizeof (Rec_Type));
    Ptr_Glob      = (Rec_Pointer *) malloc (sizeof (Rec_Type));

    Ptr_Glob->Ptr_Comp                = Next_Ptr_Glob;
    Ptr_Glob->Discr                   = Ident_1;
    Ptr_Glob->variant.var_1.Enum_Comp = Ident_3;
    Ptr_Glob->variant.var_1.Int_Comp  = 40;
    strcpy (Ptr_Glob->variant.var_1.Str_Comp,
              "DHRYSTONE PROGRAM, SOME STRING");
    strcpy (Str_1_Loc, "DHRYSTONE PROGRAM, 1\'ST STRING");
    strcpy (System_Type, SYSTYPE);
    strcpy (Compiler_Type, COMPTYPE);

    /*
     *   Was missing in published program. Without this
```

```
      *   initialization, Ar_2_Glob [8][7] would have an
      *   undefined value.
      *   Warning: With 16-Bit processors and Number_Of_Runs > 32000,
      *   overflow may occur for this array element.
      */
     Ar_2_Glob[8][7] = 10;

     printf ("\n");
     printf ("Dhrystone Benchmark, Version 2.0 (Language: C)\n");
     printf ("\n");
     printf ("System:   %s\n", System_Type);
     printf ("Compiler: %s\n", Compiler_Type);
     printf ("\n");
     if (Reg)
     {
          printf ("Program compiled with 'register' attribute\n");
     }
     else
     {
          printf ("Program compiled without 'register' attribute\n");
     }
     printf ("\n");
     {
          int  n;

          if (argc > 1)
          {
               n = atoi (argv[1]);
          }
          else
          {
#ifdef ALCYON
               /*
                *   stupid alcyon scanf(3) is bad
                */
               n = 5000;
#else
               printf ("Please give the number of runs through the benchmark: ");
               scanf ("%d", &n);
#endif
          }
          Number_Of_Runs = n;
     }
     printf ("\n");

     printf ("Execution starts, %d runs through Dhrystone\n", Number_Of_Runs);


     /*
      *   Start timer
      */
#ifdef TIMES
     times (&time_info);
```

```
    Begin_Time = (long) time_info.tms_utime;
#endif
#ifdef TIME
    dum = (long) 0;
    Begin_Time = time (&dum);
#endif

    for (Run_Index = 1; Run_Index <= Number_Of_Runs; ++Run_Index)
    {
        /*
         *   Ch_1_Glob == 'A', Ch_2_Glob == 'B', Bool_Glob == true
         */
        Proc_5 ();
        Proc_4 ();

        Int_1_Loc = 2;
        Int_2_Loc = 3;
        strcpy (Str_2_Loc, "DHRYSTONE PROGRAM, 2'ND STRING");
        Enum_Loc  = Ident_2;

        /*
         *   Bool_Glob == 1
         */
        Bool_Glob = !Func_2 (Str_1_Loc, Str_2_Loc);
        while (Int_1_Loc < Int_2_Loc)/* loop body executed once */
        {
            /*
             *   Int_3_Loc == 7
             */
            Int_3_Loc = 5 * Int_1_Loc - Int_2_Loc;
            Proc_7 (Int_1_Loc, Int_2_Loc, &Int_3_Loc);
            Int_1_Loc += 1;
        }
        /*
         *   Int_1_Loc == 3, Int_2_Loc == 3, Int_3_Loc == 7
         */
        Proc_8 (Ar_1_Glob, Ar_2_Glob, Int_1_Loc, Int_3_Loc);

        /*
         *   Int_Glob == 5
         */
        Proc_1 (Ptr_Glob);
        for (Ch_Index = 'A'; Ch_Index <= Ch_2_Glob; ++Ch_Index)
        {
            /*
             *   loop body executed twice
             */
            if (Enum_Loc == Func_1 (Ch_Index, 'C'))
            {
                /*
                 *   then, not executed
                 */
                Proc_6 (Ident_1, &Enum_Loc);
```

```
                strcpy (Str_2_Loc, "DHRYSTONE PROGRAM, 3'RD STRING");
                Int_2_Loc = Run_Index;
                Int_Glob = Run_Index;
            }
        }
        /*
         *   Int_1_Loc == 3, Int_2_Loc == 3, Int_3_Loc == 7
         */
        Int_2_Loc = Int_2_Loc * Int_1_Loc;
        Int_1_Loc = Int_2_Loc / Int_3_Loc;
        Int_2_Loc = 7 * (Int_2_Loc - Int_3_Loc) - Int_1_Loc;
        /*
         *   Int_1_Loc == 1, Int_2_Loc == 13, Int_3_Loc == 7
         */
        Proc_2 (&Int_1_Loc);
        /*
         *   Int_1_Loc == 5
         */

    }

    /*
     *   Stop timer
     */
#ifdef TIMES
    times (&time_info);
    End_Time = (long) time_info.tms_utime;
#endif
#ifdef TIME
    dum = (long) 0;
    End_Time = time (&dum);
#endif

    printf ("Execution ends\n");
    printf ("\n");
    printf ("Final values of the variables used in the benchmark:\n");
    printf ("\n");
    printf ("Int_Glob:            %d\n", Int_Glob);
    printf ("        should be:   %d\n", 5);
    printf ("Bool_Glob:           %d\n", Bool_Glob);
    printf ("        should be:   %d\n", 1);
    printf ("Ch_1_Glob:           %c\n", Ch_1_Glob);
    printf ("        should be:   %c\n", 'A');
    printf ("Ch_2_Glob:           %c\n", Ch_2_Glob);
    printf ("        should be:   %c\n", 'B');
    printf ("Ar_1_Glob[8]:        %d\n", Ar_1_Glob[8]);
    printf ("        should be:   %d\n", 7);
    printf ("Ar_2_Glob[8][7]:     %d\n", Ar_2_Glob[8][7]);
    printf ("        should be:   Number_Of_Runs + 10\n");
    printf ("Ptr_Glob->\n");
    printf ("  Ptr_Comp:          %d\n", (int) Ptr_Glob->Ptr_Comp);
    printf ("        should be:   (implementation-dependent)\n");
    printf ("  Discr:             %d\n", Ptr_Glob->Discr);
```

```
    printf ("        should be:   %d\n", 0);
    printf ("  Enum_Comp:         %d\n", Ptr_Glob->variant.var_1.Enum_Comp);
    printf ("        should be:   %d\n", 2);
    printf ("  Int_Comp:          %d\n", Ptr_Glob->variant.var_1.Int_Comp);
    printf ("        should be:   %d\n", 17);
    printf ("  Str_Comp:          %s\n", Ptr_Glob->variant.var_1.Str_Comp);
    printf ("        should be:   DHRYSTONE PROGRAM, SOME STRING\n");
    printf ("Next_Ptr_Glob->\n");
    printf ("  Ptr_Comp:          %d\n", (int) Next_Ptr_Glob->Ptr_Comp);
    printf ("        should be:   (implementation-dependent), same as above\n");
    printf ("  Discr:             %d\n", Next_Ptr_Glob->Discr);
    printf ("        should be:   %d\n", 0);
    printf ("  Enum_Comp:         %d\n", Next_Ptr_Glob->variant.var_1.Enum_Comp);
    printf ("        should be:   %d\n", 1);
    printf ("  Int_Comp:          %d\n", Next_Ptr_Glob->variant.var_1.Int_Comp);
    printf ("        should be:   %d\n", 18);
    printf ("  Str_Comp:          %s\n",
            Next_Ptr_Glob->variant.var_1.Str_Comp);
    printf ("        should be:   DHRYSTONE PROGRAM, SOME STRING\n");
    printf ("Int_1_Loc:           %d\n", Int_1_Loc);
    printf ("        should be:   %d\n", 5);
    printf ("Int_2_Loc:           %d\n", Int_2_Loc);
    printf ("        should be:   %d\n", 13);
    printf ("Int_3_Loc:           %d\n", Int_3_Loc);
    printf ("        should be:   %d\n", 7);
    printf ("Enum_Loc:            %d\n", Enum_Loc);
    printf ("        should be:   %d\n", 1);
    printf ("Str_1_Loc:           %s\n", Str_1_Loc);
    printf ("        should be:   DHRYSTONE PROGRAM, 1'ST STRING\n");
    printf ("Str_2_Loc:           %s\n", Str_2_Loc);
    printf ("        should be:   DHRYSTONE PROGRAM, 2'ND STRING\n");
    printf ("\n");

    User_Time = End_Time - Begin_Time;

    if (User_Time < Too_Small_Time)
    {
        printf ("Measured time too small to obtain meaningful results\n");
        printf ("Please increase number of runs\n");
        printf ("\n");
    }
    else
    {
#ifdef TIME
        Microseconds          = (float) User_Time * Usec_Per_Second
                              / (float) Number_Of_Runs;
        Dhrystones_Per_Second = (float) Number_Of_Runs
                              / (float) User_Time;
#else
        Microseconds          = (float) User_Time * Usec_Per_Second
                              / ((float) HZ * ((float) Number_Of_Runs));
        Dhrystones_Per_Second = ((float) HZ * (float) Number_Of_Runs)
                              / (float) User_Time;
```

```
#endif
          printf ("Microseconds for one run through Dhrystone: ");
          printf ("%6.1f \n", Microseconds);
          printf ("Dhrystones per Second:                      ");
          printf ("%6.1f \n", Dhrystones_Per_Second);
          printf ("\n");
     }


     exit (0);
}


Proc_1 (Ptr_Val_Par)             /* executed once */
REG Rec_Pointer *Ptr_Val_Par;
{
     /*
      *   Local variable, initialized with Ptr_Val_Par->Ptr_Comp,
      *   corresponds to "rename" in Ada, "with" in Pascal
      */
     REG Rec_Pointer    *Next_Record = Ptr_Val_Par->Ptr_Comp;
     /* == Ptr_Glob_Next */

     str_asgn (*Ptr_Val_Par->Ptr_Comp, *Ptr_Glob);
     Ptr_Val_Par->variant.var_1.Int_Comp = 5;
     Next_Record->variant.var_1.Int_Comp
          = Ptr_Val_Par->variant.var_1.Int_Comp;
     Next_Record->Ptr_Comp = Ptr_Val_Par->Ptr_Comp;
     Proc_3 (&(Next_Record->Ptr_Comp));
     /*
      *   Ptr_Val_Par->Ptr_Comp->Ptr_Comp == Ptr_Glob->Ptr_Comp
      */
     if (Next_Record->Discr == Ident_1)
     {
          /*
           *   then, executed
           */
          Next_Record->variant.var_1.Int_Comp = 6;
          Proc_6 (Ptr_Val_Par->variant.var_1.Enum_Comp,
                   &Next_Record->variant.var_1.Enum_Comp);
          Next_Record->Ptr_Comp = Ptr_Glob->Ptr_Comp;
          Proc_7 (Next_Record->variant.var_1.Int_Comp, 10,
                   &Next_Record->variant.var_1.Int_Comp);
     }
     else
          /*
           *   not executed
           */
          str_asgn (*Ptr_Val_Par, *Ptr_Val_Par->Ptr_Comp);
}


Proc_2 (Int_Par_Ref)             /* executed once */
```

```
One_Fifty      *Int_Par_Ref;         /* *Int_Par_Ref == 1, becomes 4 */
{
     One_Fifty Int_Loc;
     Enumeration   Enum_Loc;

     Int_Loc = *Int_Par_Ref + 10;
     do
     {
          /*
           *   executed once
           */
          if (Ch_1_Glob == 'A')
          {
               /*
                *   then, executed
                */
               Int_Loc -= 1;
               *Int_Par_Ref = Int_Loc - Int_Glob;
               Enum_Loc = Ident_1;
          }

     } while (Enum_Loc != Ident_1);   /* true */
}


Proc_3 (Ptr_Ref_Par)             /* executed once */
Rec_Pointer **Ptr_Ref_Par;       /* Ptr_Ref_Par becomes Ptr_Glob */
{
     if (Ptr_Glob != Null)
          /*
           *   then, executed
           */
          *Ptr_Ref_Par = Ptr_Glob->Ptr_Comp;
     else
          /*
           *   not executed
           */
          Int_Glob = 100;

     Proc_7 (10, Int_Glob, &Ptr_Glob->variant.var_1.Int_Comp);
}



Proc_4 ()              /* executed once */
{
     Boolean Bool_Loc;

     Bool_Loc = Ch_1_Glob == 'A';
     Bool_Glob = Bool_Loc | Bool_Glob;
     Ch_2_Glob = 'B';
}
```

```
Proc_5 ()               /* executed once */
{
     Ch_1_Glob = 'A';
     Bool_Glob = false;
}


#ifdef  NOSTRUCTASSIGN
/*
 *   Procedure for the assignment of structures,
 *   if the C compiler doesn't support this feature
 */
memcpy (d, s, l)
register char  *d;
register char  *s;
register int    l;
{
     while (l--)
          *d++ = *s++;
}
#endif
@//E*O*F dhry_p1.c//
chmod u=rw,g=r,o=r dhry_p1.c

echo x - dhry_p2.c
sed 's/^@//' > "dhry_p2.c" <<'@//E*O*F dhry_p2.c//'
/*
 **************************************************************************
 *
 *                   "DHRYSTONE" Benchmark Program
 *                   -----------------------------
 *
 *  Version:    C, Version 2.0
 *
 *  File:       dhry_pack_2.c (part 3 of 3)
 *
 *  Date:       March 3, 1988
 *
 *  Author:     Reinhold P. Weicker
 *
 **************************************************************************
 */

#include "dhry_glo.h"

#ifndef REG
#define REG             /* REG becomes defined as empty */
#endif                  /* i.e. no register variables   */

extern int      Int_Glob;
extern char     Ch_1_Glob;
```

```
Proc_6 (Enum_Val_Par, Enum_Ref_Par)  /* executed once */
Enumeration   Enum_Val_Par;      /* Enum_Val_Par == Ident_3 */
Enumeration    *Enum_Ref_Par;         /* Enum_Ref_Par becomes Ident_2 */
{
     *Enum_Ref_Par = Enum_Val_Par;
     if (!Func_3 (Enum_Val_Par))
          /*
           *   then, not executed
           */
          *Enum_Ref_Par = Ident_4;

     switch (Enum_Val_Par)
     {
     case Ident_1:
          *Enum_Ref_Par = Ident_1;
          break;
     case Ident_2:
          if (Int_Glob > 100)
               *Enum_Ref_Par = Ident_1;
          else
               *Enum_Ref_Par = Ident_4;
          break;
     case Ident_3:               /* executed */
          *Enum_Ref_Par = Ident_2;
          break;
     case Ident_4:
          break;
     case Ident_5:
          *Enum_Ref_Par = Ident_3;
          break;
     }
}


Proc_7 (Int_1_Par_Val, Int_2_Par_Val, Int_Par_Ref) /* executed three times */
One_Fifty Int_1_Par_Val;
One_Fifty Int_2_Par_Val;
One_Fifty      *Int_Par_Ref;
/*
 *   first call:      Int_1_Par_Val == 2, Int_2_Par_Val == 3,
 *                    Int_Par_Ref becomes 7
 *   second call:     Int_1_Par_Val == 10, Int_2_Par_Val == 5,
 *                    Int_Par_Ref becomes 17
 *   third call:      Int_1_Par_Val == 6, Int_2_Par_Val == 10,
 *                    Int_Par_Ref becomes 18
 */
{
     One_Fifty Int_Loc;

     Int_Loc = Int_1_Par_Val + 2;
     *Int_Par_Ref = Int_2_Par_Val + Int_Loc;
}
```

```
Proc_8 (Ar_1_Par_Ref, Ar_2_Par_Ref, Int_1_Par_Val, Int_2_Par_Val)/* executed once
*/
Array    Ar_1_Par_Ref[];
Array    Ar_2_Par_Ref[][50];
int     Int_1_Par_Val;              /* Int_Par_Val_1 == 3 */
int     Int_2_Par_Val;              /* Int_Par_Val_2 == 7 */
{
     REG One_Fifty Int_Index;
     REG One_Fifty     Int_Loc;

     Int_Loc = Int_1_Par_Val + 5;
     Ar_1_Par_Ref[Int_Loc] = Int_2_Par_Val;
     Ar_1_Par_Ref[Int_Loc + 1] = Ar_1_Par_Ref[Int_Loc];
     Ar_1_Par_Ref[Int_Loc + 30] = Int_Loc;
     for (Int_Index = Int_Loc; Int_Index <= Int_Loc + 1; ++Int_Index)
          Ar_2_Par_Ref[Int_Loc][Int_Index] = Int_Loc;
     Ar_2_Par_Ref[Int_Loc][Int_Loc - 1] += 1;
     Ar_2_Par_Ref[Int_Loc + 20][Int_Loc] = Ar_1_Par_Ref[Int_Loc];
     Int_Glob = 5;
}


Enumeration Func_1 (Ch_1_Par_Val, Ch_2_Par_Val) /* executed three times */
/*
 *   first call:      Ch_1_Par_Val == 'H', Ch_2_Par_Val == 'R'
 *   second call:     Ch_1_Par_Val == 'A', Ch_2_Par_Val == 'C'
 *   third call:      Ch_1_Par_Val == 'B', Ch_2_Par_Val == 'C'
 */
Capital_Letter    Ch_1_Par_Val;
Capital_Letter    Ch_2_Par_Val;
{
     Capital_Letter    Ch_1_Loc;
     Capital_Letter    Ch_2_Loc;

     Ch_1_Loc = Ch_1_Par_Val;
     Ch_2_Loc = Ch_1_Loc;
     if (Ch_2_Loc != Ch_2_Par_Val)
          /*
           *   then, executed
           */
          return (Ident_1);
     else               /* not executed */
     {
          Ch_1_Glob = Ch_1_Loc;
          return (Ident_2);
     }
}


Boolean Func_2 (Str_1_Par_Ref, Str_2_Par_Ref)  /* executed once */
/*
```

```
 *   Str_1_Par_Ref == "DHRYSTONE PROGRAM, 1'ST STRING"
 *   Str_2_Par_Ref == "DHRYSTONE PROGRAM, 2'ND STRING"
 */
char    Str_1_Par_Ref[31];
char    Str_2_Par_Ref[31];
{
     REG One_Thirty     Int_Loc;
     Capital_Letter     Ch_Loc;

     Int_Loc = 2;
     while (Int_Loc <= 2)        /* loop body executed once */
          if (Func_1 (Str_1_Par_Ref[Int_Loc],
                    Str_2_Par_Ref[Int_Loc + 1]) == Ident_1)
          {
               /*
                *   then, executed
                */
               Ch_Loc = 'A';
               Int_Loc += 1;
          }
     if (Ch_Loc >= 'W' && Ch_Loc < 'Z')
          /*
           *   then, not executed
           */
          Int_Loc = 7;
     if (Ch_Loc == 'R')
          /*
           *   then, not executed
           */
          return (true);
     else
     {
          /*
           *   executed
           */
          if (strcmp (Str_1_Par_Ref, Str_2_Par_Ref) > 0)
          {
               /*
                *   then, not executed
                */
               Int_Loc += 7;
               Int_Glob = Int_Loc;
               return (true);
          }
          else
               /*
                *   executed
                */
               return (false);
     }
}
```

```
Boolean Func_3 (Enum_Par_Val)             /* executed once */
Enumeration   Enum_Par_Val;          /* Enum_Par_Val == Ident_3 */
{
     Enumeration   Enum_Loc;

     Enum_Loc = Enum_Par_Val;
     if (Enum_Loc == Ident_3)
          /*
           *   then, executed
           */
          return (true);
}
@//E*O*F dhry_p2.c//
chmod u=rw,g=r,o=r dhry_p2.c

echo Inspecting for damage in transit...
temp=/tmp/shar$$; dtemp=/tmp/.shar$$
trap "rm -f $temp $dtemp; exit" 0 1 2 3 15
cat > $temp <<\!!!
      10      74     496 MANIFEST
      56     126     935 Makefile
     344    1908   16178 README
      65     192    1773 OUT.ST
      43     107     712 config.h
     107     260    2189 dhry_glo.h
     447    1294   10993 dhry_p1.c
     196     535    4456 dhry_p2.c
    1268    4496   37732 total
!!!
wc  MANIFEST Makefile README OUT.ST config.h dhry_glo.h dhry_p1.c dhry_p2.c | sed
's=[^ ]*/==' | diff -b $temp - >$dtemp
if [ -s $dtemp ]
then echo "Ouch [diff of wc output]:" ; cat $dtemp
else echo "No problems found."
fi
exit 0


From 340063300436457988
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,48f4ff5a07658c56,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,48f4ff5a07658c56,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-12-09 12:07:42 PST
Xref: utzoo comp.sys.atari.st:23254 comp.sources.atari.st:268
Path: utzoo!utgpu!jarvis.csri.toronto.edu!clyde.concordia.ca!uunet!cs.utexas.edu!
sun-barr!newstop!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sys.atari.st,comp.sources.atari.st
Subject: v03INF3: Change to posting format
Message-ID: <34935@grapevine.uucp>
Date: 9 Dec 89 20:07:42 GMT
```

Expires: 8 Jan 90 20:07:42 GMT
Sender: news@grapevine.uucp
Reply-To: koreth@panarthea.ebay.sun.com (Steven Grimm)
Followup-To: comp.sys.atari.st
Lines: 43
Approved: koreth@panarthea.ebay.sun.com (Steven Grimm)

The votes are in on changing from shar to uue-{zoo,arc} format, and the
uues have it.

I'll try to regularly (once per volume) post pointers to source code for
the various archiving and decoding programs, and make sure that they're
available on the panarthea archive.

Some notable comments:

>From: tim@brspyr1.brs.com (Tim Northrup)
>
>Both ZOO and ARC are now commonly available on Unix systems (much more so
>than when comp.sources.atari.st was started), so why bother with a shar
>file?

>From: Ron Abramson <rona@hpdml93.hp.com>
>
>It seems to me that if uuencoded arc files are good enough for
>comp.binaries.atari.st, then they ought to be good enough for
>comp.sources.atari.st
>
>I don't understand why a person (be it a he or she :~}) should need
>to learn a new technique to retrieve source files.

>From: sluka@stoch.fmi.uni-passau.de (Bernd Sluka)
>
>I always un-shar your sources and repack them as arc- or zoo-archives
>on my sun before transfering them to the st via an PC-clone.
>So in my point of view it would be better to use uue'd arc- or zoo-files
>for the sources, too.

And one of the few dissenting votes,

>From: saj@chinet.chi.il.us (Stephen Jacobs)
>
>I like to keep sources in readable form. What you might do is dictate a
>particular shar format, and provide tools to create and disassemble those
>files in the monthly posting.

"   !" - Marcel Marceau
Steven Grimm   Moderator, comp.{sources,binaries}.atari.st
koreth@ebay.sun.com   ...!sun!ebay!koreth

From -8023194126288584050

X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,56a8c8b031cd58a0,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1989-12-21 08:47:00 PST
Path: utzoo!utgpu!jarvis.csri.toronto.edu!clyde.concordia.ca!uunet!cs.utexas.edu!
sun-barr!newstop!grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i013: dhrystone -- Dhrystone benchmark program
Keywords: shar, C
Message-ID: <34993@grapevine.uucp>
Date: 21 Dec 89 16:47:00 GMT
Sender: news@grapevine.uucp
Reply-To: rosenkra@hall.cray.com (Bill Rosenkranz)
Lines: 1348
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: rosenkra@hall.cray.com (Bill Rosenkranz)
Posting-number: Volume 3, Issue 13
Archive-name: dhrystone

this is a port of the dhrystone bm, v2.0, i found on the MIPS BBS. when
run on my mega4, compiled with alcyon 4.14, it ran at about 719. see
the file OUT.ST for the results of that run.

-bill
net: rosenkra@hall.cray.com
CIS: 71460,17

--------cut here--------------cut here----------------cut here---------------
This is a shell archive. Remove anything before this line,
then unpack it by saving it in a file and typing "sh file".
#
Wrapped by hall!rosenkra on Tue Dec 5 23:34:28 CST 1989
Contents: MANIFEST Makefile README OUT.ST config.h dhry_glo.h dhry_p1.c
dhry_p2.c

```
echo x - MANIFEST
sed 's/^@//' > "MANIFEST" <<'@//E*O*F MANIFEST//'
total 66
-rw-r--r--  1 rosenkra      550 Dec  5 23:30 MANIFEST
-rw-r--r--  1 rosenkra      935 Dec  5 23:22 Makefile
-rw-r--r--  1 rosenkra     1773 Dec  5 23:22 OUT.ST
-rw-r--r--  1 rosenkra    16178 Dec  5 23:22 README
-rw-r--r--  1 rosenkra      235 Dec  5 23:30 README.ST
-rw-r--r--  1 rosenkra      712 Dec  5 23:22 config.h
-rw-r--r--  1 rosenkra     2189 Dec  5 23:22 dhry_glo.h
-rw-r--r--  1 rosenkra    10993 Dec  5 23:22 dhry_p1.c
-rw-r--r--  1 rosenkra     4456 Dec  5 23:22 dhry_p2.c
@//E*O*F MANIFEST//
chmod u=rw,g=r,o=r MANIFEST

echo x - Makefile
```

```
sed 's/^@//' > "Makefile" <<'@//E*O*F Makefile//'
# makefile for Dhrystone v2.0 for atariST (alcyon)
#
# first edit config.h for your system/compiler...
#
UUENCODE  = c:\bin\uue.ttp

TARGET        = dhry.ttp
ARCFILE       = dhry20st.arc
SHARFILE  = dhry20st.shr
UUFILE        = dhry.uue
SRCS      = config.h dhry_glo.h dhry_p1.c dhry_p2.c
OTHERS        = readme makefile
DISTFILES= $(OTHERS) $(SRCS)
OBJS      = dhry_p1.o dhry_p2.o
LIBS      = -llibc
LDFLAGS       = -s -new
CFLAGS        = -new


# main target...
#
$(TARGET):    $(OBJS)
          $(CC) $(LDFLAGS) -o $(TARGET) $(OBJS) $(LIBS)


# to run it...
#
run:      $(TARGET)
          $(TARGET) 30000

clean:
          rm $(OBJS) errs

clobber:  clean
          rm $(TARGET)

arc:
          $(ARC) a $(ARCFILE) $(DISTFILES)
          @echo done making $(ARCFILE)

uue:
          $(UUENCODE) $(TARGET)
          @echo done making $(UUFILE)

shar:
          $(SHAR) $(DISTFILES) >$(SHARFILE)
          @echo done making $(SHARFILE)

# dependencies...
#
dhry_glo.h:   config.h
          $(TOUCH) dhry_glo.h
```

```
dhry_p1.o:    dhry_glo.h dhry_p1.c
dhry_p2.o:    dhry_glo.h dhry_p2.c

@//E*O*F Makefile//
chmod u=rw,g=r,o=r Makefile

echo x - README
sed 's/^@//' > "README" <<'@//E*O*F README//'
/* NOTE: edit config.h first for your system!!! */

/*
 ****************************************************************************
 *
 *                   "DHRYSTONE" Benchmark Program
 *                   -----------------------------
 *
 *  Version:    C, Version 2.0
 *
 *  Date:       March 3, 1988
 *
 *  Author:     Reinhold P. Weicker
 *                      Siemens AG, E STE 35
 *                      Postfach 3240
 *                      8520 Erlangen
 *                      Germany (West)
 *                              Phone:  [xxx-49]-9131-7-20330
 *                                      (8-17 Central European Time)
 *                              Usenet: ..!mcvax!unido!estevax!weicker
 *
 *              Original Version (in Ada) published in
 *              "Communications of the ACM" vol. 27., no. 10 (Oct. 1984),
 *              pp. 1013 - 1030, together with the statistics
 *              on which the distribution of statements etc. is based.
 *
 *              In this C version, the following C library functions are used:
 *              - strcpy, strcmp (inside the measurement loop)
 *              - printf, scanf (outside the measurement loop)
 *              In addition, UNIX system calls "times ()" or "time ()"
 *              are used for execution time measurement. For measurements
 *              on other systems, these calls have to be changed.
 *
 *  Collection of Results:
 *              Reinhold Weicker (address see above) and
 *
 *              Rick Richardson
 *              PC Research. Inc.
 *              94 Apple Orchard Drive
 *              Tinton Falls, NJ 07724
 *                      Phone:  (201) 834-1378 (9-17 EST)
 *                      Usenet: ...!seismo!uunet!pcrat!rick
 *
 *      Please send results to Reinhold Weicker and/or Rick Richardson.
```

```
 *      Complete information should be given on hardware and software used.
 *      Hardware information includes: Machine type, CPU, type and size
 *      of caches; for microprocessors: clock frequency, memory speed
 *      (number of wait states).
 *      Software information includes: Compiler (and runtime library)
 *      manufacturer and version, compilation switches, OS version.
 *      The Operating System version may give an indication about the
 *      compiler; Dhrystone itself performs no OS calls in the measurement loop.
 *
 *      The complete output generated by the program should be mailed
 *      such that at least some checks for correctness can be made.
 */
/*
 ************************************************************************
 *
 *  History:    This version C/2.0 has been made for two reasons:
 *
 *              1) There is an obvious need for a common C version of
 *              Dhrystone, since C is at present the most popular system
 *              programming language for the class of processors
 *              (microcomputers, minicomputers) where Dhrystone is used most.
 *              There should be, as far as possible, only one C version of
 *              Dhrystone such that results can be compared without
 *              restrictions. In the past, the C versions distributed
 *              by Rick Richardson (Version 1.1) and by Reinhold Weicker
 *              had small (though not significant) differences.
 *
 *              2) As far as it is possible without changes to the Dhrystone
 *              statistics, optimizing compilers should be prevented from
 *              removing significant statements.
 *
 *              This C version has been developed in cooperation with
 *              Rick Richardson (Tinton Falls, NJ), it incorporates many
 *              ideas from the "Version 1.1" distributed previously by
 *              him over the UNIX network Usenet.
 *              I also thank Chaim Benedelac (National Semiconductor),
 *              David Ditzel (SUN), Earl Killian and John Mashey (MIPS),
 *              Alan Smith and Rafael Saavedra-Barrera (UC at Berkeley)
 *              for their help with comments on earlier versions of the
 *              benchmark.
 *
 *  Changes:    In the initialization part, this version follows mostly
 *              Rick Richardson's version distributed via Usenet, not the
 *              version distributed earlier via floppy disk by Reinhold Weicker.
 *              As a concession to older compilers, names have been made
 *              unique within the first 8 characters.
 *              Inside the measurement loop, this version follows the
 *              version previously distributed by Reinhold Weicker.
 *
 *              At several places in the benchmark, code has been added,
 *              but within the measurement loop only in branches that
 *              are not executed. The intention is that optimizing compilers
 *              should be prevented from moving code out of the measurement
```

```
 *              loop, or from removing code altogether. Since the statements
 *              that are executed within the measurement loop have NOT been
 *              changed, the numbers defining the "Dhrystone distribution"
 *              (distribution of statements, operand types and locality)
 *              still hold. Except for sophisticated optimizing compilers,
 *              execution times for this version should be the same as
 *              for previous versions.
 *
 *              Since it has proven difficult to subtract the time for the
 *              measurement loop overhead in a correct way, the loop check
 *              has been made a part of the benchmark. This does have
 *              an impact - though a very minor one - on the distribution
 *              statistics which have been updated for this version.
 *
 *              All changes within the measurement loop are described
 *              and discussed in the companion paper "Rationale for
 *              Dhrystone version 2".
 *
 *              Because of the self-imposed limitation that the order and
 *              distribution of the executed statements should not be
 *              changed, there are still cases where optimizing compilers
 *              may not generate code for some statements. To a certain
 *              degree, this is unavoidable for small synthetic benchmarks.
 *              Users of the benchmark are advised to check code listings
 *              whether code is generated for all statements of Dhrystone.
 */
/*
 ************************************************************************
 *
 * Defines:     The following "Defines" are possible:
 *              -DREG=register          (default: Not defined)
 *                      As an approximation to what an average C programmer
 *                      might do, the "register" storage class is applied
 *                      (if enabled by -DREG=register)
 *                      - for local variables, if they are used (dynamically)
 *                        five or more times
 *                      - for parameters if they are used (dynamically)
 *                        six or more times
 *                      Note that an optimal "register" strategy is
 *                      compiler-dependent, and that "register" declarations
 *                      do not necessarily lead to faster execution.
 *              -DNOSTRUCTASSIGN        (default: Not defined)
 *                      Define if the C compiler does not support
 *                      assignment of structures.
 *              -DNOENUMS               (default: Not defined)
 *                      Define if the C compiler does not support
 *                      enumeration types.
 *              -DTIMES                 (default)
 *              -DTIME
 *                      The "times" function of UNIX (returning process times)
 *                      or the "time" function (returning wallclock time)
 *                      is used for measurement.
 *                      For single user machines, "time ()" is adequate. For
```

```
 *                      multi-user machines where you cannot get single-user
 *                      access, use the "times ()" function. If you have
 *                      neither, use a stopwatch in the dead of night.
 *                      "printf"s are provided marking the points "Start Timer"
 *                      and "Stop Timer". DO NOT use the UNIX "time(1)"
 *                      command, as this will measure the total time to
 *                      run this program, which will (erroneously) include
 *                      the time to allocate storage (malloc) and to perform
 *                      the initialization.
 *              -DHZ=nnn                (default: 60)
 *                      The function "times" returns process times in
 *                      1/HZ seconds, with HZ = 60 for most systems.
 *                      CHECK YOUR SYSTEM DESCRIPTION BEFORE YOU JUST APPLY
 *                      THE DEFAULT VALUE.
 */
/*
 *************************************************************************
 *
 *  Compilation model and measurement (IMPORTANT):
 *
 *  This C version of Dhrystone consists of three files:
 *  - dhry_global.h (this file, containing global definitions and comments)
 *  - dhry_pack_1.c (containing the code corresponding to Ada package Pack_1)
 *  - dhry_pack_2.c (containing the code corresponding to Ada package Pack_2)
 *
 *  The following "ground rules" apply for measurements:
 *  - Separate compilation
 *  - No procedure merging
 *  - Otherwise, compiler optimizations are allowed but should be indicated
 *  - Default results are those without register declarations
 *  See the companion paper "Rationale for Dhrystone Version 2" for a more
 *  detailed discussion of these ground rules.
 *
 *  For 16-Bit processors (e.g. 80186, 80286), times for all compilation
 *  models ("small", "medium", "large" etc.) should be given if possible,
 *  together with a definition of these models for the compiler system used.
 */
/*
 *************************************************************************
 *
 *  Dhrystone (C version) statistics:
 *
 *  [Comment from the first distribution, updated for version 2.
 *   Note that because of language differences, the numbers are slightly
 *   different from the Ada version.]
 *
 *  The following program contains statements of a high level programming
 *  language (here: C) in a distribution considered representative:
 *
 *    assignments                   52 (51.0 %)
 *    control statements            33 (32.4 %)
 *    procedure, function calls     17 (16.7 %)
 *
```

```
*  103 statements are dynamically executed. The program is balanced with
*  respect to the three aspects:
*
*    - statement type
*    - operand type
*    - operand locality
*         operand global, local, parameter, or constant.
*
*  The combination of these three aspects is balanced only approximately.
*
*  1. Statement Type:
*  -----------------              number
*
*     V1 = V2                      9
*       (incl. V1 = F(..)
*     V = Constant                12
*     Assignment,                  7
*       with array element
*     Assignment,                  6
*       with record component
*                                 --
*                                 34       34
*
*     X = Y +|-|"&&"|"|" Z         5
*     X = Y +|-|"==" Constant      6
*     X = X +|- 1                  3
*     X = Y *|/ Z                  2
*     X = Expression,              1
*           two operators
*     X = Expression,              1
*           three operators
*                                 --
*                                 18       18
*
*     if ....                     14
*       with "else"      7
*       without "else"   7
*           executed        3
*           not executed    4
*     for ...                      7  |  counted every time
*     while ...                    4  |  the loop condition
*     do ... while                 1  |  is evaluated
*     switch ...                   1
*     break                        1
*     declaration with             1
*       initialization
*                                 --
*                                 34       34
*
*     P (...)  procedure call     11
*       user procedure      10
*       library procedure    1
*     X = F (...)
```

```
*             function  call      6
*       user function        5
*       library function     1
*                              --
*                              17       17
*                                      ---
*                                      103
*
*    The average number of parameters in procedure or function calls
*    is 1.82 (not counting the function values as implicit parameters).
*
*
*  2. Operators
*  ------------
*                           number    approximate
*                                     percentage
*
*    Arithmetic              32          50.8
*
*       +                      21          33.3
*       -                       7          11.1
*       *                       3           4.8
*       / (int div)             1           1.6
*
*    Comparison              27           42.8
*
*       ==                      9           14.3
*       /=                      4           6.3
*       >                       1           1.6
*       <                       3           4.8
*       >=                      1           1.6
*       <=                      9           14.3
*
*    Logic                    4            6.3
*
*       && (AND-THEN)           1           1.6
*       |  (OR)                1           1.6
*       !  (NOT)               2           3.2
*
*                            --          -----
*                            63          100.1
*
*
*  3. Operand Type (counted once per operand reference):
*  ---------------
*                           number    approximate
*                                     percentage
*
*     Integer               175        72.3 %
*     Character              45        18.6 %
*     Pointer                12         5.0 %
*     String30                6         2.5 %
*     Array                   2         0.8 %
```

```
 *     Record                    2         0.8 %
 *                             ---       -------
 *                             242       100.0 %
 *
 *  When there is an access path leading to the final operand (e.g. a record
 *  component), only the final data type on the access path is counted.
 *
 *
 *  4. Operand Locality:
 *  -------------------
 *                                 number    approximate
 *                                           percentage
 *
 *     local variable               114        47.1 %
 *     global variable               22         9.1 %
 *     parameter                     45        18.6 %
 *     value                         23         9.5 %
 *     reference                     22         9.1 %
 *     function result                6         2.5 %
 *     constant                      55        22.7 %
 *                                  ---       -------
 *                                  242       100.0 %
 *
 *
 *  The program does not compute anything meaningful, but it is syntactically
 *  and semantically correct. All variables have a value assigned to them
 *  before they are used as a source operand.
 *
 *  There has been no explicit effort to account for the effects of a
 *  cache, or to balance the use of long or short displacements for code or
 *  data.
 *
 ****************************************************************************
 */

@//E*O*F README//
chmod u=rw,g=r,o=r README

echo x - OUT.ST
sed 's/^@//' > "OUT.ST" <<'@//E*O*F OUT.ST//'
    dhry.ttp 30000

Dhrystone Benchmark, Version 2.0 (Language: C)

System:   Atari MegsST 4 (TOS, 8 Mhz)
Compiler: DRI/Alcyon 4.14

Program compiled with 'register' attribute


Execution starts, 30000 runs through Dhrystone
Execution ends
```

```
Final values of the variables used in the benchmark:

Int_Glob:            5
        should be:   5
Bool_Glob:           1
        should be:   1
Ch_1_Glob:           A
        should be:   A
Ch_2_Glob:           B
        should be:   B
Ar_1_Glob[8]:        7
        should be:   7
Ar_2_Glob[8][7]:     30010
        should be:   Number_Of_Runs + 10
Ptr_Glob->
  Ptr_Comp:          22250
        should be:   (implementation-dependent)
  Discr:             0
        should be:   0
  Enum_Comp:         2
        should be:   2
  Int_Comp:          17
        should be:   17
  Str_Comp:          DHRYSTONE PROGRAM, SOME STRING
        should be:   DHRYSTONE PROGRAM, SOME STRING
Next_Ptr_Glob->
  Ptr_Comp:          22250
        should be:   (implementation-dependent), same as above
  Discr:             0
        should be:   0
  Enum_Comp:         1
        should be:   1
  Int_Comp:          18
        should be:   18
  Str_Comp:          DHRYSTONE PROGRAM, SOME STRING
        should be:   DHRYSTONE PROGRAM, SOME STRING
Int_1_Loc:           5
        should be:   5
Int_2_Loc:           13
        should be:   13
Int_3_Loc:           7
        should be:   7
Enum_Loc:            1
        should be:   1
Str_1_Loc:           DHRYSTONE PROGRAM, 1'ST STRING
        should be:   DHRYSTONE PROGRAM, 1'ST STRING
Str_2_Loc:           DHRYSTONE PROGRAM, 2'ND STRING
        should be:   DHRYSTONE PROGRAM, 2'ND STRING

Microseconds for one run through Dhrystone: 1449.9
Dhrystones per Second:                       719.2

@//E*O*F OUT.ST//
```

```
chmod u=rw,g=r,o=r OUT.ST

echo x - config.h
sed 's/^@//' > "config.h" <<'@//E*O*F config.h//'
/*
 *   config.h - configuration for Dhrystone 2.0
 *
 *   choices:
 *        -DREG=register
 *              -DNOSTRUCTASSIGN
 *              -DNOENUMS
 *              -DTIMES
 *              -DTIME
 *              -DHZ=nnn
 */

/*
 *   atariST (TOS, dri/alcyon 4.14)
 *
 *   int is 16 bits here. so max times thru is < 32k times.
 *
 */
#define ALCYON
#define SYSTYPE         "Atari MegsST 4 (TOS 1.2, 8 Mhz 68000)"
#define COMPTYPE   "DRI/Alcyon 4.14 (16-bit ints)"
#define REG        register
#define TIME
#define NOENUM

/*
 *   BSD
 *
#define SYSTYPE         "BSD Unix"
#define COMPTYPE   "pcc"
#define REG        register
#define TIME
 */

/*
 *   SYSV
 *
#define SYSTYPE         "System V.3 Unix"
#define COMPTYPE   "pcc"
#define REG        register
#define TIME
 */

@//E*O*F config.h//
chmod u=rw,g=r,o=r config.h

echo x - dhry_glo.h
sed 's/^@//' > "dhry_glo.h" <<'@//E*O*F dhry_glo.h//'
#include "config.h"
```

```
/*
 ***************************************************************************
 *
 *                   "DHRYSTONE" Benchmark Program
 *                   -----------------------------
 *
 *  Version:    C, Version 2.0
 *
 *  File:       dhry_glo.h (part 1 of 3)
 *
 *  Date:       March 3, 1988
 *
 *  Author:     Reinhold P. Weicker
 *
 ***************************************************************************
 */

/*
 *   Compiler and system dependent definitions:
 */
#ifndef TIME
#ifndef TIMES          /* Use "times" function for measurement */
#define TIMES          /* unless explicitly defined otherwise  */
#endif
#endif

#ifndef HZ
#define HZ         60   /* Use HZ = 60 for "times" function     */
#endif                  /* unless explicitly defined otherwise  */

#ifdef TIMES
#include <sys/types.h>
#include <sys/times.h>      /* for "times" */
#endif

#define Usec_Per_Second 1000000.0/* UNIX C returns process times in seconds/HZ */

#ifdef  NOSTRUCTASSIGN
#define str_asgn(d,s)   memcpy(&(d), &(s), sizeof(d))
#else
#define str_asgn(d,s)   d = s
#endif

/*
 *   for boolean and enumeration types in Ada, Pascal
 */
#ifdef  NOENUM
#define Ident_1         0
#define Ident_2         1
#define Ident_3         2
#define Ident_4         3
#define Ident_5         4
```

```
#define Enumeration    int
#else
typedef enum {Ident_1, Ident_2, Ident_3, Ident_4, Ident_5} Enumeration;
#endif


/*
 *   General definitions:
 */
#include <stdio.h>          /* for strcpy, strcmp */

#define Null       0        /* Value of a Null pointer */
#define true       1
#define false      0
#define One_Thirty int
#define One_Fifty  int
#define Capital_Letter  char
#define Boolean         int
#define Str_30          char
#define Array      int
#define Rec_Type   struct record
#define Rec_Pointer     struct recort

struct record
{
     struct record            *Ptr_Comp;
     Enumeration              Discr;

     union
     {
          struct
          {
               Enumeration   Enum_Comp;
               int       Int_Comp;
               char      Str_Comp[31];

          }              var_1;
          struct
          {
               Enumeration   E_Comp_2;
               char      Str_2_Comp[31];

          }              var_2;
          struct
          {
               char      Ch_1_Comp;
               char      Ch_2_Comp;

          }              var_3;

     }                   variant;
};
```

```
@//E*O*F dhry_glo.h//
chmod u=rw,g=r,o=r dhry_glo.h

echo x - dhry_p1.c
sed 's/^@//' > "dhry_p1.c" <<'@//E*O*F dhry_p1.c//'
/*
 ************************************************************************
 *
 *                   "DHRYSTONE" Benchmark Program
 *                   -----------------------------
 *
 *  Version:    C, Version 2.0
 *
 *  File:       dhry_pack_1.c (part 2 of 3)
 *
 *  Date:       March 3, 1988
 *
 *  Author:     Reinhold P. Weicker
 *
 ************************************************************************
 */

#include "dhry_glo.h"

/*
 *   Global Variables:
 */
Rec_Pointer    *Ptr_Glob,
           *Next_Ptr_Glob;
int      Int_Glob;
Boolean  Bool_Glob;
char     Ch_1_Glob,
         Ch_2_Glob;
int      Ar_1_Glob[50];
int      Ar_2_Glob[50][50];
char     System_Type[80];
char     Compiler_Type[80];

/*
 *   forward declaration necessary since Enumeration may not simply be int
 */
extern char    *malloc ();
Enumeration   Func_1 ();

#ifndef REG
Boolean  Reg = false;       /* REG becomes defined as empty */
#define REG                 /* i.e. no register variables   */
#else
Boolean  Reg = true;
#endif

/*
 *   variables for time measurement:
```

```
 */
#ifdef TIMES
struct tms      time_info;
extern int      times ();       /* see library function "times" */
#endif
#ifdef TIME
extern long     time ();        /* see library function "time"  */
#endif

#define Too_Small_Time  2       /* Measurements should last at least 2
                           seconds */

long     Begin_Time,
            End_Time,
         User_Time;
float    Microseconds,
            Dhrystones_Per_Second;



main (argc, argv)
int     argc;
char   *argv[];
{

/*
 *   main program, corresponds to procedures Main and Proc_0 in the
 *   Ada version
 */

    One_Fifty Int_1_Loc;
    REG One_Fifty      Int_2_Loc;
    One_Fifty     Int_3_Loc;
    REG char        Ch_Index;
    Enumeration   Enum_Loc;
    Str_30        Str_1_Loc[31];
    Str_30        Str_2_Loc[31];
    REG int   Run_Index;
    REG int   Number_Of_Runs;
    long     dum = 0L;


    /*
     *   Initializations
     */
    Next_Ptr_Glob = (Rec_Pointer *) malloc (sizeof (Rec_Type));
    Ptr_Glob      = (Rec_Pointer *) malloc (sizeof (Rec_Type));

    Ptr_Glob->Ptr_Comp                = Next_Ptr_Glob;
    Ptr_Glob->Discr                   = Ident_1;
    Ptr_Glob->variant.var_1.Enum_Comp = Ident_3;
    Ptr_Glob->variant.var_1.Int_Comp  = 40;
    strcpy (Ptr_Glob->variant.var_1.Str_Comp,
```

```
              "DHRYSTONE PROGRAM, SOME STRING");
     strcpy (Str_1_Loc, "DHRYSTONE PROGRAM, 1\'ST STRING");
     strcpy (System_Type, SYSTYPE);
     strcpy (Compiler_Type, COMPTYPE);

     /*
      *   Was missing in published program. Without this
      *   initialization, Ar_2_Glob [8][7] would have an
      *   undefined value.
      *   Warning: With 16-Bit processors and Number_Of_Runs > 32000,
      *   overflow may occur for this array element.
      */
     Ar_2_Glob[8][7] = 10;

     printf ("\n");
     printf ("Dhrystone Benchmark, Version 2.0 (Language: C)\n");
     printf ("\n");
     printf ("System:   %s\n", System_Type);
     printf ("Compiler: %s\n", Compiler_Type);
     printf ("\n");
     if (Reg)
     {
          printf ("Program compiled with 'register' attribute\n");
     }
     else
     {
          printf ("Program compiled without 'register' attribute\n");
     }
     printf ("\n");
     {
          int  n;

          if (argc > 1)
          {
               n = atoi (argv[1]);
          }
          else
          {
#ifdef ALCYON
               /*
                *   stupid alcyon scanf(3) is bad
                */
               n = 5000;
#else
               printf ("Please give the number of runs through the benchmark: ");
               scanf ("%d", &n);
#endif
          }
          Number_Of_Runs = n;
     }
     printf ("\n");

     printf ("Execution starts, %d runs through Dhrystone\n", Number_Of_Runs);
```

```
    /*
     *   Start timer
     */
#ifdef TIMES
    times (&time_info);
    Begin_Time = (long) time_info.tms_utime;
#endif
#ifdef TIME
    dum = (long) 0;
    Begin_Time = time (&dum);
#endif

    for (Run_Index = 1; Run_Index <= Number_Of_Runs; ++Run_Index)
    {
        /*
         *   Ch_1_Glob == 'A', Ch_2_Glob == 'B', Bool_Glob == true
         */
        Proc_5 ();
        Proc_4 ();

        Int_1_Loc = 2;
        Int_2_Loc = 3;
        strcpy (Str_2_Loc, "DHRYSTONE PROGRAM, 2'ND STRING");
        Enum_Loc  = Ident_2;

        /*
         *   Bool_Glob == 1
         */
        Bool_Glob = !Func_2 (Str_1_Loc, Str_2_Loc);
        while (Int_1_Loc < Int_2_Loc)/* loop body executed once */
        {
            /*
             *   Int_3_Loc == 7
             */
            Int_3_Loc = 5 * Int_1_Loc - Int_2_Loc;
            Proc_7 (Int_1_Loc, Int_2_Loc, &Int_3_Loc);
            Int_1_Loc += 1;
        }
        /*
         *   Int_1_Loc == 3, Int_2_Loc == 3, Int_3_Loc == 7
         */
        Proc_8 (Ar_1_Glob, Ar_2_Glob, Int_1_Loc, Int_3_Loc);

        /*
         *   Int_Glob == 5
         */
        Proc_1 (Ptr_Glob);
        for (Ch_Index = 'A'; Ch_Index <= Ch_2_Glob; ++Ch_Index)
        {
            /*
             *   loop body executed twice
```

```
              */
             if (Enum_Loc == Func_1 (Ch_Index, 'C'))
             {
                  /*
                   *   then, not executed
                   */
                  Proc_6 (Ident_1, &Enum_Loc);
                  strcpy (Str_2_Loc, "DHRYSTONE PROGRAM, 3'RD STRING");
                  Int_2_Loc = Run_Index;
                  Int_Glob = Run_Index;
             }
        }
        /*
         *   Int_1_Loc == 3, Int_2_Loc == 3, Int_3_Loc == 7
         */
        Int_2_Loc = Int_2_Loc * Int_1_Loc;
        Int_1_Loc = Int_2_Loc / Int_3_Loc;
        Int_2_Loc = 7 * (Int_2_Loc - Int_3_Loc) - Int_1_Loc;
        /*
         *   Int_1_Loc == 1, Int_2_Loc == 13, Int_3_Loc == 7
         */
        Proc_2 (&Int_1_Loc);
        /*
         *   Int_1_Loc == 5
         */

    }

    /*
     *   Stop timer
     */
#ifdef TIMES
    times (&time_info);
    End_Time = (long) time_info.tms_utime;
#endif
#ifdef TIME
    dum = (long) 0;
    End_Time = time (&dum);
#endif

    printf ("Execution ends\n");
    printf ("\n");
    printf ("Final values of the variables used in the benchmark:\n");
    printf ("\n");
    printf ("Int_Glob:            %d\n", Int_Glob);
    printf ("        should be:   %d\n", 5);
    printf ("Bool_Glob:           %d\n", Bool_Glob);
    printf ("        should be:   %d\n", 1);
    printf ("Ch_1_Glob:           %c\n", Ch_1_Glob);
    printf ("        should be:   %c\n", 'A');
    printf ("Ch_2_Glob:           %c\n", Ch_2_Glob);
    printf ("        should be:   %c\n", 'B');
    printf ("Ar_1_Glob[8]:        %d\n", Ar_1_Glob[8]);
```

```
    printf ("        should be:   %d\n", 7);
    printf ("Ar_2_Glob[8][7]:    %d\n", Ar_2_Glob[8][7]);
    printf ("        should be:   Number_Of_Runs + 10\n");
    printf ("Ptr_Glob->\n");
    printf ("  Ptr_Comp:          %d\n", (int) Ptr_Glob->Ptr_Comp);
    printf ("        should be:   (implementation-dependent)\n");
    printf ("  Discr:             %d\n", Ptr_Glob->Discr);
    printf ("        should be:   %d\n", 0);
    printf ("  Enum_Comp:         %d\n", Ptr_Glob->variant.var_1.Enum_Comp);
    printf ("        should be:   %d\n", 2);
    printf ("  Int_Comp:          %d\n", Ptr_Glob->variant.var_1.Int_Comp);
    printf ("        should be:   %d\n", 17);
    printf ("  Str_Comp:          %s\n", Ptr_Glob->variant.var_1.Str_Comp);
    printf ("        should be:   DHRYSTONE PROGRAM, SOME STRING\n");
    printf ("Next_Ptr_Glob->\n");
    printf ("  Ptr_Comp:          %d\n", (int) Next_Ptr_Glob->Ptr_Comp);
    printf ("        should be:   (implementation-dependent), same as above\n");
    printf ("  Discr:             %d\n", Next_Ptr_Glob->Discr);
    printf ("        should be:   %d\n", 0);
    printf ("  Enum_Comp:         %d\n", Next_Ptr_Glob->variant.var_1.Enum_Comp);
    printf ("        should be:   %d\n", 1);
    printf ("  Int_Comp:          %d\n", Next_Ptr_Glob->variant.var_1.Int_Comp);
    printf ("        should be:   %d\n", 18);
    printf ("  Str_Comp:          %s\n",
              Next_Ptr_Glob->variant.var_1.Str_Comp);
    printf ("        should be:   DHRYSTONE PROGRAM, SOME STRING\n");
    printf ("Int_1_Loc:           %d\n", Int_1_Loc);
    printf ("        should be:   %d\n", 5);
    printf ("Int_2_Loc:           %d\n", Int_2_Loc);
    printf ("        should be:   %d\n", 13);
    printf ("Int_3_Loc:           %d\n", Int_3_Loc);
    printf ("        should be:   %d\n", 7);
    printf ("Enum_Loc:            %d\n", Enum_Loc);
    printf ("        should be:   %d\n", 1);
    printf ("Str_1_Loc:           %s\n", Str_1_Loc);
    printf ("        should be:   DHRYSTONE PROGRAM, 1'ST STRING\n");
    printf ("Str_2_Loc:           %s\n", Str_2_Loc);
    printf ("        should be:   DHRYSTONE PROGRAM, 2'ND STRING\n");
    printf ("\n");

    User_Time = End_Time - Begin_Time;

    if (User_Time < Too_Small_Time)
    {
        printf ("Measured time too small to obtain meaningful results\n");
        printf ("Please increase number of runs\n");
        printf ("\n");
    }
    else
    {
#ifdef TIME
        Microseconds          = (float) User_Time * Usec_Per_Second
                      / (float) Number_Of_Runs;
```

```
          Dhrystones_Per_Second = (float) Number_Of_Runs
                       / (float) User_Time;
#else
          Microseconds          = (float) User_Time * Usec_Per_Second
                       / ((float) HZ * ((float) Number_Of_Runs));
          Dhrystones_Per_Second = ((float) HZ * (float) Number_Of_Runs)
                       / (float) User_Time;
#endif
          printf ("Microseconds for one run through Dhrystone: ");
          printf ("%6.1f \n", Microseconds);
          printf ("Dhrystones per Second:                      ");
          printf ("%6.1f \n", Dhrystones_Per_Second);
          printf ("\n");
     }


     exit (0);
}


Proc_1 (Ptr_Val_Par)             /* executed once */
REG Rec_Pointer *Ptr_Val_Par;
{
     /*
      *   Local variable, initialized with Ptr_Val_Par->Ptr_Comp,
      *   corresponds to "rename" in Ada, "with" in Pascal
      */
     REG Rec_Pointer    *Next_Record = Ptr_Val_Par->Ptr_Comp;
     /* == Ptr_Glob_Next */

     str_asgn (*Ptr_Val_Par->Ptr_Comp, *Ptr_Glob);
     Ptr_Val_Par->variant.var_1.Int_Comp = 5;
     Next_Record->variant.var_1.Int_Comp
          = Ptr_Val_Par->variant.var_1.Int_Comp;
     Next_Record->Ptr_Comp = Ptr_Val_Par->Ptr_Comp;
     Proc_3 (&(Next_Record->Ptr_Comp));
     /*
      *   Ptr_Val_Par->Ptr_Comp->Ptr_Comp == Ptr_Glob->Ptr_Comp
      */
     if (Next_Record->Discr == Ident_1)
     {
          /*
           *   then, executed
           */
          Next_Record->variant.var_1.Int_Comp = 6;
          Proc_6 (Ptr_Val_Par->variant.var_1.Enum_Comp,
                    &Next_Record->variant.var_1.Enum_Comp);
          Next_Record->Ptr_Comp = Ptr_Glob->Ptr_Comp;
          Proc_7 (Next_Record->variant.var_1.Int_Comp, 10,
                    &Next_Record->variant.var_1.Int_Comp);
     }
     else
          /*
```

```
          *   not executed
          */
         str_asgn (*Ptr_Val_Par, *Ptr_Val_Par->Ptr_Comp);
}


Proc_2 (Int_Par_Ref)            /* executed once */
One_Fifty      *Int_Par_Ref;         /* *Int_Par_Ref == 1, becomes 4 */
{
     One_Fifty Int_Loc;
     Enumeration   Enum_Loc;

     Int_Loc = *Int_Par_Ref + 10;
     do
     {
          /*
           *   executed once
           */
          if (Ch_1_Glob == 'A')
          {
               /*
                *   then, executed
                */
               Int_Loc -= 1;
               *Int_Par_Ref = Int_Loc - Int_Glob;
               Enum_Loc = Ident_1;
          }

     } while (Enum_Loc != Ident_1);   /* true */
}


Proc_3 (Ptr_Ref_Par)            /* executed once */
Rec_Pointer **Ptr_Ref_Par;      /* Ptr_Ref_Par becomes Ptr_Glob */
{
     if (Ptr_Glob != Null)
          /*
           *   then, executed
           */
          *Ptr_Ref_Par = Ptr_Glob->Ptr_Comp;
     else
          /*
           *   not executed
           */
          Int_Glob = 100;

     Proc_7 (10, Int_Glob, &Ptr_Glob->variant.var_1.Int_Comp);
}



Proc_4 ()              /* executed once */
{
```

```
     Boolean Bool_Loc;

     Bool_Loc = Ch_1_Glob == 'A';
     Bool_Glob = Bool_Loc | Bool_Glob;
     Ch_2_Glob = 'B';
}


Proc_5 ()               /* executed once */
{
     Ch_1_Glob = 'A';
     Bool_Glob = false;
}


#ifdef  NOSTRUCTASSIGN
/*
 *   Procedure for the assignment of structures,
 *   if the C compiler doesn't support this feature
 */
memcpy (d, s, l)
register char  *d;
register char  *s;
register int    l;
{
     while (l--)
          *d++ = *s++;
}
#endif
@//E*O*F dhry_p1.c//
chmod u=rw,g=r,o=r dhry_p1.c

echo x - dhry_p2.c
sed 's/^@//' > "dhry_p2.c" <<'@//E*O*F dhry_p2.c//'
/*
 **************************************************************************
 *
 *                   "DHRYSTONE" Benchmark Program
 *                   -----------------------------
 *
 *  Version:    C, Version 2.0
 *
 *  File:       dhry_pack_2.c (part 3 of 3)
 *
 *  Date:       March 3, 1988
 *
 *  Author:     Reinhold P. Weicker
 *
 **************************************************************************
 */

#include "dhry_glo.h"
```

```
#ifndef REG
#define REG             /* REG becomes defined as empty */
#endif                  /* i.e. no register variables   */

extern int      Int_Glob;
extern char     Ch_1_Glob;


Proc_6 (Enum_Val_Par, Enum_Ref_Par)  /* executed once */
Enumeration   Enum_Val_Par;      /* Enum_Val_Par == Ident_3 */
Enumeration    *Enum_Ref_Par;         /* Enum_Ref_Par becomes Ident_2 */
{
     *Enum_Ref_Par = Enum_Val_Par;
     if (!Func_3 (Enum_Val_Par))
          /*
           *   then, not executed
           */
          *Enum_Ref_Par = Ident_4;

     switch (Enum_Val_Par)
     {
     case Ident_1:
          *Enum_Ref_Par = Ident_1;
          break;
     case Ident_2:
          if (Int_Glob > 100)
               *Enum_Ref_Par = Ident_1;
          else
               *Enum_Ref_Par = Ident_4;
          break;
     case Ident_3:               /* executed */
          *Enum_Ref_Par = Ident_2;
          break;
     case Ident_4:
          break;
     case Ident_5:
          *Enum_Ref_Par = Ident_3;
          break;
     }
}


Proc_7 (Int_1_Par_Val, Int_2_Par_Val, Int_Par_Ref) /* executed three times */
One_Fifty Int_1_Par_Val;
One_Fifty Int_2_Par_Val;
One_Fifty      *Int_Par_Ref;
/*
 *   first call:      Int_1_Par_Val == 2, Int_2_Par_Val == 3,
 *                    Int_Par_Ref becomes 7
 *   second call:     Int_1_Par_Val == 10, Int_2_Par_Val == 5,
 *                    Int_Par_Ref becomes 17
 *   third call:      Int_1_Par_Val == 6, Int_2_Par_Val == 10,
 *                    Int_Par_Ref becomes 18
```

```
 */
{
     One_Fifty Int_Loc;

     Int_Loc = Int_1_Par_Val + 2;
     *Int_Par_Ref = Int_2_Par_Val + Int_Loc;
}


Proc_8 (Ar_1_Par_Ref, Ar_2_Par_Ref, Int_1_Par_Val, Int_2_Par_Val)/* executed once
*/
Array    Ar_1_Par_Ref[];
Array    Ar_2_Par_Ref[][50];
int     Int_1_Par_Val;              /* Int_Par_Val_1 == 3 */
int     Int_2_Par_Val;              /* Int_Par_Val_2 == 7 */
{
     REG One_Fifty Int_Index;
     REG One_Fifty     Int_Loc;

     Int_Loc = Int_1_Par_Val + 5;
     Ar_1_Par_Ref[Int_Loc] = Int_2_Par_Val;
     Ar_1_Par_Ref[Int_Loc + 1] = Ar_1_Par_Ref[Int_Loc];
     Ar_1_Par_Ref[Int_Loc + 30] = Int_Loc;
     for (Int_Index = Int_Loc; Int_Index <= Int_Loc + 1; ++Int_Index)
          Ar_2_Par_Ref[Int_Loc][Int_Index] = Int_Loc;
     Ar_2_Par_Ref[Int_Loc][Int_Loc - 1] += 1;
     Ar_2_Par_Ref[Int_Loc + 20][Int_Loc] = Ar_1_Par_Ref[Int_Loc];
     Int_Glob = 5;
}


Enumeration Func_1 (Ch_1_Par_Val, Ch_2_Par_Val) /* executed three times */
/*
 *   first call:      Ch_1_Par_Val == 'H', Ch_2_Par_Val == 'R'
 *   second call:     Ch_1_Par_Val == 'A', Ch_2_Par_Val == 'C'
 *   third call:      Ch_1_Par_Val == 'B', Ch_2_Par_Val == 'C'
 */
Capital_Letter    Ch_1_Par_Val;
Capital_Letter    Ch_2_Par_Val;
{
     Capital_Letter    Ch_1_Loc;
     Capital_Letter    Ch_2_Loc;

     Ch_1_Loc = Ch_1_Par_Val;
     Ch_2_Loc = Ch_1_Loc;
     if (Ch_2_Loc != Ch_2_Par_Val)
          /*
           *   then, executed
           */
          return (Ident_1);
     else               /* not executed */
     {
          Ch_1_Glob = Ch_1_Loc;
```

```
          return (Ident_2);
     }
}


Boolean Func_2 (Str_1_Par_Ref, Str_2_Par_Ref)  /* executed once */
/*
 *   Str_1_Par_Ref == "DHRYSTONE PROGRAM, 1'ST STRING"
 *   Str_2_Par_Ref == "DHRYSTONE PROGRAM, 2'ND STRING"
 */
char    Str_1_Par_Ref[31];
char    Str_2_Par_Ref[31];
{
     REG One_Thirty     Int_Loc;
     Capital_Letter     Ch_Loc;

     Int_Loc = 2;
     while (Int_Loc <= 2)        /* loop body executed once */
          if (Func_1 (Str_1_Par_Ref[Int_Loc],
                    Str_2_Par_Ref[Int_Loc + 1]) == Ident_1)
          {
               /*
                *   then, executed
                */
               Ch_Loc = 'A';
               Int_Loc += 1;
          }
     if (Ch_Loc >= 'W' && Ch_Loc < 'Z')
          /*
           *   then, not executed
           */
          Int_Loc = 7;
     if (Ch_Loc == 'R')
          /*
           *   then, not executed
           */
          return (true);
     else
     {
          /*
           *   executed
           */
          if (strcmp (Str_1_Par_Ref, Str_2_Par_Ref) > 0)
          {
               /*
                *   then, not executed
                */
               Int_Loc += 7;
               Int_Glob = Int_Loc;
               return (true);
          }
          else
               /*
```

```
               *   executed
               */
              return (false);
     }
}


Boolean Func_3 (Enum_Par_Val)             /* executed once */
Enumeration   Enum_Par_Val;          /* Enum_Par_Val == Ident_3 */
{
     Enumeration   Enum_Loc;

     Enum_Loc = Enum_Par_Val;
     if (Enum_Loc == Ident_3)
          /*
           *   then, executed
           */
          return (true);
}
@//E*O*F dhry_p2.c//
chmod u=rw,g=r,o=r dhry_p2.c

echo Inspecting for damage in transit...
temp=/tmp/shar$$; dtemp=/tmp/.shar$$
trap "rm -f $temp $dtemp; exit" 0 1 2 3 15
cat > $temp <<\!!!
      10      74     496 MANIFEST
      56     126     935 Makefile
     344    1908   16178 README
      65     192    1773 OUT.ST
      43     107     712 config.h
     107     260    2189 dhry_glo.h
     447    1294   10993 dhry_p1.c
     196     535    4456 dhry_p2.c
    1268    4496   37732 total
!!!
wc  MANIFEST Makefile README OUT.ST config.h dhry_glo.h dhry_p1.c dhry_p2.c | sed
's=[^ ]*/==' | diff -b $temp - >$dtemp
if [ -s $dtemp ]
then echo "Ouch [diff of wc output]:" ; cat $dtemp
else echo "No problems found."
fi
exit 0


From 7205299607958601710
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,c00f098610d65058,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1990-02-04 04:15:35 PST
Path: gmdzi!unido!mcsun!sunic!uupsi!rpi!brutus.cs.uiuc.edu!apple!sun-barr!newstop!
grapevine!koreth@panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
```

Newsgroups: comp.sources.atari.st
Subject: v03i025: barcodes -- Metafont file for printing barcodes
Keywords: Metafont
Message-ID: <35222@grapevine.uucp>
Date: 3 Feb 90 19:47:39 GMT
Sender: news@grapevine.uucp
Reply-To: csrobe@cs.wm.edu (Chip Roberson)
Lines: 291
Approved: koreth@panarthea.ebay.sun.com
Posted: Sat Feb 3 20:47:39 1990

Submitted-by: csrobe@cs.wm.edu (Chip Roberson)
Posting-number: Volume 3, Issue 25
Archive-name: barcodes

I noticed that someone was asking about barcodes recently and in
my metafont searches for [La]Tex, I discovered the barcodes.mf. I
know nothing more that what it says in the header and I have not
tried it. If you decide to use this you will need TeX (or LaTeX)
and metafont (available from archive-server@panarthea.ebay.sun.com).

barcodes.mf can be anonymously ftp'd from sun.soe.clarkson.edu's
pub/tex-fonts directory.

cheers,
-chip
-=- Charles S. Roberson   ARPANET: csrobe@cs.wm.edu   -=-
-=- VA Remote Sensing Study   UUCP: ...!uunet!cs.wm.edu!csrobe -=-
-=- Dept of Comp. Sci.   Compu$erve: 71500.2056@compuserve.com -=-
-=- College of William and Mary Ma Bell: (804) 221-3455 [W]   -=-
-=- Williamsburg, VA 23185   (804) 229-5530 [H]   -=-

                    Fur: The look that kills.

---Begin barcodes.mf---
% 10-Nov-87 14:05:55-PST,10055;000000000000
% Return-Path: <CLDLV%NASAGISS.BITNET@forsythe.stanford.edu>
% Date: Tue, 10 Nov 87 15:46:49 EDT
% From: Dimitri Vulis <CLDLV%NASAGISS.BITNET@forsythe.stanford.edu>
% Subject: Foo (bar codes)
%
% Dear Malcolm: I believe that the following file would of general
% interest, but I think it's too long for the digest:
%
% I was amazed to find out that business people pay enormous money for the
% ability to produce bar codes. This ought to be TRIVIAL with \TeX.
% So, I got hold of a public domain BASIC program that supposedly does that and
% lifted the codes and put them into METAFONT (it was EASY). Caveat(s): I have
% never tested these codes with an OCR. The BASIC program said it used `3 of 9'
% encoding. I presume it's not the same as UPC. The sizes may be off---I
% took them from the BASIC program that used HP LJ's 100dpi graphics mode.
% Disclaimer: I haven't tested these codes. Use them at your own risk.
% Remarks:

```
% 1) White space is a displayable character.
% 2) There are 9 significant strips, bwbwbwbwb; 3 of them are wide.
% 3) The white_naro; at the end of each code is the inter-code spacing.
% Credits: the original BASIC program was written by `Bill Wood Mil., WI' and
% later re-written by `Bill Baines, Enfield, CT.'. METAFONT version is by
% `Dimitri Vulis CLDLV@NASAGISS.BITNET'
mode_setup;
font_identifier "DVBar"; font_size .35in#;
naro#:=.01in#;                % narrow strip width
wide#:=3naro#;                % wide strip width
wdth#:=3wide#+7naro#;         % common width of all codes
hght#:=.35in#;                % common height of all codes
define_pixels(naro,wide,wdth,hght);
def white_naro=
  wid:=wid+naro;
enddef;
def black_naro=
   filldraw (wid,0)--(wid,hght)--(wid+naro,hght)--(wid+naro,0)--cycle;
  wid:=wid+naro;
enddef;
def white_wide=
  wid:=wid+wide;
enddef;
def black_wide=
   filldraw (wid,0)--(wid,hght)--(wid+wide,hght)--(wid+wide,0)--cycle;
  wid:=wid+wide;
enddef;
beginchar("1",wdth#,hght#,0);
wid:=0;
black_wide; white_naro; black_naro; white_wide; black_naro;
white_naro; black_naro; white_naro; black_wide; white_naro;
endchar;
beginchar("2",wdth#,hght#,0);
wid:=0;
black_naro; white_naro; black_wide; white_wide; black_naro;
white_naro; black_naro; white_naro; black_wide; white_naro;
endchar;
beginchar("3",wdth#,hght#,0);
wid:=0;
black_wide; white_naro; black_wide; white_wide; black_naro;
white_naro; black_naro; white_naro; black_naro; white_naro;
endchar;
beginchar("4",wdth#,hght#,0);
wid:=0;
black_naro; white_naro; black_naro; white_wide; black_wide;
white_naro; black_naro; white_naro; black_wide; white_naro;
endchar;
beginchar("5",wdth#,hght#,0);
wid:=0;
black_wide; white_naro; black_naro; white_wide; black_wide;
white_naro; black_naro; white_naro; black_naro; white_naro;
endchar;
beginchar("6",wdth#,hght#,0);
```

```
wid:=0;
black_naro; white_naro; black_wide; white_wide; black_wide;
white_naro; black_naro; white_naro; black_naro; white_naro;
endchar;
beginchar("7",wdth#,hght#,0);
wid:=0;
black_naro; white_naro; black_naro; white_wide; black_naro;
white_naro; black_wide; white_naro; black_wide; white_naro;
endchar;
beginchar("8",wdth#,hght#,0);
wid:=0;
black_wide; white_naro; black_naro; white_wide; black_naro;
white_naro; black_wide; white_naro; black_naro; white_naro;
endchar;
beginchar("9",wdth#,hght#,0);
wid:=0;
black_naro; white_naro; black_wide; white_wide; black_naro;
white_naro; black_wide; white_naro; black_naro; white_naro;
endchar;
beginchar("0",wdth#,hght#,0);
wid:=0;
black_naro; white_naro; black_naro; white_wide; black_wide;
white_naro; black_wide; white_naro; black_naro; white_naro;
endchar;
beginchar("A",wdth#,hght#,0);
wid:=0;
black_wide; white_naro; black_naro; white_naro; black_naro;
white_wide; black_naro; white_naro; black_wide; white_naro;
endchar;
beginchar("B",wdth#,hght#,0);
wid:=0;
black_naro; white_naro; black_wide; white_naro; black_naro;
white_wide; black_naro; white_naro; black_wide; white_naro;
endchar;
beginchar("C",wdth#,hght#,0);
wid:=0;
black_wide; white_naro; black_wide; white_naro; black_naro;
white_wide; black_naro; white_naro; black_naro; white_naro;
endchar;
beginchar("D",wdth#,hght#,0);
wid:=0;
black_naro; white_naro; black_naro; white_naro; black_wide;
white_wide; black_naro; white_naro; black_wide; white_naro;
endchar;
beginchar("E",wdth#,hght#,0);
wid:=0;
black_wide; white_naro; black_naro; white_naro; black_wide;
white_wide; black_naro; white_naro; black_naro; white_naro;
endchar;
beginchar("F",wdth#,hght#,0);
wid:=0;
black_naro; white_naro; black_wide; white_naro; black_wide;
white_wide; black_naro; white_naro; black_naro; white_naro;
```

```
endchar;
beginchar("G",wdth#,hght#,0);
wid:=0;
black_naro; white_naro; black_naro; white_naro; black_naro;
white_wide; black_wide; white_naro; black_wide; white_naro;
endchar;
beginchar("H",wdth#,hght#,0);
wid:=0;
black_wide; white_naro; black_naro; white_naro; black_naro;
white_wide; black_wide; white_naro; black_naro; white_naro;
endchar;
beginchar("I",wdth#,hght#,0);
wid:=0;
black_naro; white_naro; black_wide; white_naro; black_naro;
white_wide; black_wide; white_naro; black_naro; white_naro;
endchar;
beginchar("J",wdth#,hght#,0);
wid:=0;
black_naro; white_naro; black_naro; white_naro; black_wide;
white_wide; black_wide; white_naro; black_naro; white_naro;
endchar;
beginchar("K",wdth#,hght#,0);
wid:=0;
black_wide; white_naro; black_naro; white_naro; black_naro;
white_naro; black_naro; white_wide; black_wide; white_naro;
endchar;
beginchar("L",wdth#,hght#,0);
wid:=0;
black_naro; white_naro; black_wide; white_naro; black_naro;
white_naro; black_naro; white_wide; black_wide; white_naro;
endchar;
beginchar("M",wdth#,hght#,0);
wid:=0;
black_wide; white_naro; black_wide; white_naro; black_naro;
white_naro; black_naro; white_wide; black_naro; white_naro;
endchar;
beginchar("N",wdth#,hght#,0);
wid:=0;
black_naro; white_naro; black_naro; white_naro; black_wide;
white_naro; black_naro; white_wide; black_wide; white_naro;
endchar;
beginchar("O",wdth#,hght#,0);
wid:=0;
black_wide; white_naro; black_naro; white_naro; black_wide;
white_naro; black_naro; white_wide; black_naro; white_naro;
endchar;
beginchar("P",wdth#,hght#,0);
wid:=0;
black_naro; white_naro; black_wide; white_naro; black_wide;
white_naro; black_naro; white_wide; black_naro; white_naro;
endchar;
beginchar("Q",wdth#,hght#,0);
wid:=0;
```

```
black_naro; white_naro; black_naro; white_naro; black_naro;
white_naro; black_wide; white_wide; black_wide; white_naro;
endchar;
beginchar("R",wdth#,hght#,0);
wid:=0;
black_wide; white_naro; black_naro; white_naro; black_naro;
white_naro; black_wide; white_wide; black_naro; white_naro;
endchar;
beginchar("S",wdth#,hght#,0);
wid:=0;
black_naro; white_naro; black_wide; white_naro; black_naro;
white_naro; black_wide; white_wide; black_naro; white_naro;
endchar;
beginchar("T",wdth#,hght#,0);
wid:=0;
black_naro; white_naro; black_naro; white_naro; black_wide;
white_naro; black_wide; white_wide; black_naro; white_naro;
endchar;
beginchar("U",wdth#,hght#,0);
wid:=0;
black_wide; white_wide; black_naro; white_naro; black_naro;
white_naro; black_naro; white_naro; black_wide; white_naro;
endchar;
beginchar("V",wdth#,hght#,0);
wid:=0;
black_naro; white_wide; black_wide; white_naro; black_naro;
white_naro; black_naro; white_naro; black_wide; white_naro;
endchar;
beginchar("W",wdth#,hght#,0);
wid:=0;
black_wide; white_wide; black_wide; white_naro; black_naro;
white_naro; black_naro; white_naro; black_naro; white_naro;
endchar;
beginchar("X",wdth#,hght#,0);
wid:=0;
black_naro; white_wide; black_naro; white_naro; black_wide;
white_naro; black_naro; white_naro; black_wide; white_naro;
endchar;
beginchar("Y",wdth#,hght#,0);
wid:=0;
black_wide; white_wide; black_naro; white_naro; black_wide;
white_naro; black_naro; white_naro; black_naro; white_naro;
endchar;
beginchar("Z",wdth#,hght#,0);
wid:=0;
black_naro; white_wide; black_wide; white_naro; black_wide;
white_naro; black_naro; white_naro; black_naro; white_naro;
endchar;
beginchar("-",wdth#,hght#,0);          % minus
wid:=0;
black_naro; white_wide; black_naro; white_naro; black_naro;
white_naro; black_wide; white_naro; black_wide; white_naro;
endchar;
```

```
beginchar(".",wdth#,hght#,0);          %period
wid:=0;
black_wide; white_wide; black_naro; white_naro; black_naro;
white_naro; black_wide; white_naro; black_naro; white_naro;
endchar;
beginchar(" ",wdth#,hght#,0);          %white space
wid:=0;
black_naro; white_wide; black_wide; white_naro; black_naro;
white_naro; black_wide; white_naro; black_naro; white_naro;
endchar;
beginchar("*",wdth#,hght#,0);          %ascii asterisk
wid:=0;
black_naro; white_wide; black_naro; white_naro; black_wide;
white_naro; black_wide; white_naro; black_naro; white_naro;
endchar;
beginchar("$",wdth#,hght#,0);          %dollar sign
wid:=0;
black_naro; white_wide; black_naro; white_wide; black_naro;
white_wide; black_naro; white_naro; black_naro; white_naro;
endchar;
beginchar("/",wdth#,hght#,0);          %forward slash
wid:=0;
black_naro; white_wide; black_naro; white_wide; black_naro;
white_naro; black_naro; white_wide; black_naro; white_naro;
endchar;
beginchar("+",wdth#,hght#,0);          %plus
wid:=0;
black_naro; white_wide; black_naro; white_naro; black_naro;
white_wide; black_naro; white_wide; black_naro; white_naro;
endchar;
beginchar("%",wdth#,hght#,0);          %percent sign
wid:=0;
black_naro; white_naro; black_naro; white_wide; black_naro;
white_wide; black_naro; white_wide; black_naro; white_naro;
endchar;
end;     %all done
---End barcodes.mf---


From -192748174430438915
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,3c035e767ff3012,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1990-05-30 21:32:46 PST
Path: gmdzi!unido!mcsun!uunet!ogicse!ucsd!usc!cs.utexas.edu!sun-barr!newstop!male!
panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i057: gfa-uucode -- Uuencode/decode program in GFA BASIC
Keywords: basic
Message-ID: <2066@male.EBay.Sun.COM>
Date: 31 May 90 04:32:46 GMT
Sender: news@male.EBay.Sun.COM
```

Reply-To: wscart01@ulkyvx.bitnet (Stuart Carter)
Lines: 490
Approved: koreth@panarthea.ebay.sun.com
Posted: Thu May 31 05:32:46 1990

Submitted-by: wscart01@ulkyvx.bitnet (Stuart Carter)
Posting-number: Volume 3, Issue 57
Archive-name: gfa-uucode

 Here is a UUE encoder and decoder. This is for mainly the people
who don't have a UUE untility. Simply capture this file in your
buffer and later load it into GFA basic and edit out the non-program
and run. This will create a UUCODE.ACC which is a UUE accesory that
will allow you access to the UUE world. I cant take credit for this
accesory, Moches Branner made about two years ago, and no one has yet
inproved on it's speed.

Stuart Carter
University Of Louisville
Physics Dept.

```
'---------------------------CUT HERE---------------------------------
'                Stuart Carter's   Hex-decoder                      '
'              Version 1.0               4/10/90                    '
' ----------------------------------------------------------------- '
' This program is a simple Hex decoder that contains Moshse Branner's
' (Sorry if i misspelled it) UUCODE acessory. This program will help
' you get started with using file transfers that require ASCII codeings.
' Useing UUCODE.ACC is simple and straight forward except for that:
' 1) Check to make sure UUE file has been re-assembled into one smooth
'    flow of data. (I use Tempus to put files back together.)
' 2) Some UUE files dont have a little "u" or such to end a line. This
'    causes loss of data if end character is a space. Manualy insert
'    spaces if you have a file like this. (Better yet, write a program
'    to do it for you.)
'
'                                                 Have Fun, Stuart
READ bytes
DIM code%(bytes/2)
code_adr%=V:code%(0)
get_data:
READ line$
FOR line=1 TO LEN(line$) STEP 2
  byte%=VAL("&h"+MID$(line$,line,2))
  POKE code_adr%+byte_num%,byte%
  INC byte_num%
NEXT line
PRINT AT(70,1);bytes-byte_num%;"   "
IF byte_num%<bytes
  GOTO get_data
ENDIF
BSAVE "UUCODE.ACC",code_adr%,bytes
```

```
END
DATA 13583
DATA 601A00003340000001A60000187200000000000000000000000004EF900
DATA 0000364EF9000014304EF90000126C4EF9000014CA4EF9000013844EF90000
DATA 14164EF9000013064EF9000012C64EF9000032644BFAFFFE0C6D4EF9FFFA66
DATA 045D8D60F49BFC00000100206D0018226D0018D3ED001C202D001453806F06
DATA 132051C8FFFC206D0010202D001C5380421851C8FFFC286D0010D9ED001C29
DATA 4DFFFC4FECFFFC2A6D00084EAD00304EBA002A3F3C00004EBA0D5A48E7FFFE
DATA 3F3C00413F3C00024E41588F203C000F4240538066FC4CDF7FFF4E754E56FF
DATA EC4EBA0B523D40FFFE486C00003F2EFFFE4EBA0BAE5C8F3D40FFFC486EFFEC
DATA 4EBA0B88588F0C6E0028FFEC660000064EBA000C6000FFE64E5E4E754E754E
DATA 56FFFE486C000A3F3C00034EBA0B945C8F3D40FFFE302EFFFE60000002B07C
DATA 000167000006600000028486EFFFE3F3C00024EBA0B8C5C8F4EBA00FA486EFF
DATA FE42674EBA0B7C5C8F60000386000000EB07C000267000006600000028486E
DATA FFFE3F3C00024EBA0B5A5C8F4EBA04EC486EFFFE42674EBA0B4A5C8F600000
DATA 06600000024E5E4E754E56FF983F3C00194EBA1BE6548FD07C00411D40FFAE
DATA 422EFFAF486C003E486EFFAE4EBA2D16508F2F2E000C486EFF9E4EBA2DD250
DATA 8F426EFF98486EFF98486EFF9E486EFFAE4EBA0B10DEFC000C0C6E0000FF98
DATA 6600000A303C00006000004C3F3C002A486EFFAE4EBA2DDC5C8F2D40FF9A20
DATA 6EFF9A4210486EFFAE2F2E00084EBA2D82508F486EFF9E2F2E00084EBA2CAA
DATA 508F486EFF9E2F2E000C4EBA2D66508F302EFF98600000024E5E4E754E56FE
DATA D2422EFF70486EFF70486EFF804EBAFF40508F3D40FF5E0C6E0000FF5E6600
DATA 000A303C000060000110486C0044486EFF804EBA15AA508F2D40FED60C8000
DATA 0000006600001848 6C00483F3C00014EBA0A1C5C8F303C0000600000DE486E
DATA FF70486EFF604EBA2CEE508F3F3C002E486EFF704EBA2D225C8F2D40FEDA20
DATA 6EFEDA4210486C0066486EFF704EBA2BFE508F486EFF70486EFEDE4EBAFEB6
DATA 508F3D40FF5E0C6E0000FF5E6600000A303C000060000086486C006C486EFE
DATA DE4EBA1520508F2D40FED20C8000000000066000018486C006E3F3C00014EBA
DATA 09925C8F303C000060000054486EFF5E3F3C00024EBA099A5C8F486EFF6048
DATA 6C00902F2EFED24EBA20A8DEFC000C2F2EFED22F2EFED64EBA002A508F486C
DATA 009E2F2EFED24EBA208A508F2F2EFED64EBA0ADE588F2F2EFED24EBA0AD458
DATA 8F4E5E4E754E56FFAC3F3C002D486EFFB02F2E00084EBA0254DEFC000A3D40
DATA FFAC206E000C536800043028000 40C4000006D000024302EFFACC07C003FD0
DATA 7C0020206E000C22105290204110804880C07C00FF6000001A2F2E000C302E
DATA FFACC07C003FD07C00203F004EBA0CA45C8F426EFFAE6000001C2F2E000C41
DATA EEFFB0D0EEFFAE41D02F084EBA0038508F566EFFAE302EFFAEB06EFFAC6D00
DATA FFDC2F2E000C486C00A44EBA1FF8508F0C6E0000FFAC6E0000066000000660
DATA 00FF4E4E5E4E754E56FFF8206E000810104880E4403D40FFFE206E00081010
DATA 4880E940C07C0030206E000812800014881E841C27C000F80413D40FFFC20
DATA 6E000810280001 4880E540C07C003C206E000812800024881EC41C27C0003
DATA 80413D40FFFA206E000810280002 4880C07C003F3D40FFF8206E000C536800
DATA 04302800040C4000006D000024302EFFFEC07C003FD07C0020206E000C2210
DATA 5290204110804880C07C00FF6000001A2F2E000C302EFFFEC07C003FD07C00
DATA 203F004EBA0B965C8F206E000C536800043028000 40C4000006D000024302E
DATA FFFCC07C003FD07C0020206E000C22105290204110804880C07C00FF600000
DATA 1A2F2E000C302EFFFCC07C003FD07C00203F004EBA0B485C8F206E000C5368
DATA 0004302800040C4000006D000024302EFFFAC07C003FD07C0020206E000C22
DATA 1052902041108048 0C07C00FF6000001A2F2E000C302EFFFAC07C003FD07C
DATA 00203F004EBA0AFA5C8F206E000C536800043028000 40C4000006D00002430
DATA 2EFFF8C07C003FD07C0020206E000C22105290204110804880C07C00FF6000
DATA 001A2F2E000C302EFFF8C07C003FD07C00203F004EBA0AAC5C8F4E5E4E754E
DATA 56FFFC426EFFFC6000005E206E000853680004302800040C4000006D000018
DATA 206E000820105290204010104880C07C00FF6000000C2F2E00084EBA093858
```

```
DATA 8F3D40FFFE0C6EFFFFFFFE6600000A302EFFFC60000028302EFFFE206E000C
DATA D0EEFFFC1080526EFFFC302EFFFCB06E00106D00FF9A302E0010600000024E
DATA 5E4E754E56FD42422EFF4C486EFF4C486EFFAE4EBAFB1C508F3D40FD420C6E
DATA 0000FD426600000A303C000060000156486C00A8486EFFAE4EBA1186508F2D
DATA 40FD480C800000000066000018486C00AA3F3C00014EBA05F85C8F303C0000
DATA 600001242F2EFD483F3C00FF486EFE4C4EBA12E6DEFC000A0C800000000066
DATA 00000A303C0000600000FE3F3C0006486C00CC486EFE4C4EBA2852DEFC000A
DATA 4A406600FFC2486EFF4C486EFD4C486EFD4C486C00D4486EFE4C4EBA2798DE
DATA FC0014486EFF4C486EFF5C4EBAFA6A508F3D40FD420C6E0000FD426600000A
DATA 303C0000600000A4486C00DE486EFF5C4EBA10D4508F2D40FD440C80000000
DATA 0066000018486C00E23F3C00014EBA05465C8F303C000060000072486EFD42
DATA 3F3C00024EBA054E5C8F2F2EFD442F2EFD484EBA005C508F2F2EFD483F3C01
DATA 00486EFE4C4EBA1218DEFC000A0C800000000067000016486C0106486EFE4C
DATA 4EBA274E508F4A4067000010486C010C3F3C00014EBA04E25C8F2F2EFD484E
DATA BA0674588F2F2EFD444EBA066A588F4E5E4E754E56FF9A2F2E00083F3C0050
DATA 486EFFB04EBA11BCDEFC000A0C800000000066000018486C01283F3C00014E
DATA BA049A5C8F303C0000600000C4102EFFB04880907C0020C07C003F3D40FFA6
DATA 0C6E0000FFA66E000006600000A4486EFFB04EBA2768588FB06EFFA66C0000
DATA 5C3F3C000A486EFFB04EBA277C5C8F2D40FFA8206EFFA84210426EFFA46000
DATA 0010486C0144486EFFB04EBA2650508F486EFFB04EBA2728588F3D40FF9A30
DATA 2EFFA6906EFF9A53403D40FF9A302EFFA4526EFFA4B06EFF9A6F00FFC841EE
DATA FFB12D48FFAC6000001E3F2EFFA62F2E000C2F2EFFAC4EBA0020DEFC000A58
DATA AEFFAC576EFFA60C6E0000FFA66E00FFDC6000FF0C4E5E4E754E56FFFA206E
DATA 000810104880907C0020C07C003FE540206E000812280001488192 7C0020C2
DATA 7C003FE84180413D40FFFE206E000810280001488090 7C0020C07C003FE940
DATA 206E000812280002488192 7C0020C27C003FE44180413D40FFFC206E000810
DATA 2800024880907C0020C07C003FED40206E000812280003488192 7C0020C27C
DATA 003F80413D40FFFA0C6E000100106D00003E206E000C53680004302800040C
DATA 4000006D00001C302EFFFE206E000C2210529020411080488 0C07C00FF6000
DATA 00102F2E000C3F2EFFFE4EBA06D65C8F0C6E000200106D00003E206E000C53
DATA 680004302800040C4000006D00001C302EFFFC206E000C2210529020411080
DATA 4880C07C00FF600000102F2E000C3F2EFFFC4EBA06905C8F0C6E000300106D
DATA 00003E206E000C53680004302800040C4000006D00001C302EFFFA206E000C
DATA 2210529020411080488 0C07C00FF600000102F2E000C3F2EFFFA4EBA064A5C
DATA 8F4E5E4E752949E7F6294AE7F2222F0004303C00C84E42226CE7F6246CE7F2
DATA 4E75000100020101020101000101020101010101000000000000000000000001
DATA 0000010003050005050000010102010010070102010000000000000000000000
DATA 0101010201010201010201010101020101010000000000000000000000000002
DATA 0101010101060101040101010301020101040201080101000000000000000101
DATA 0109010101010101010000050103030102020100000000000000000000000000
DATA 0000000000000000000000000000000000000000000000000000004030008
DATA 0300060100080100080100040101030101000500010101000500000101 0001
DATA 0100000000000000000000000000000000000000000000000000202000000
DATA 0000000000000000000000000000000000000000000000000501000501 0001
DATA 0100010100020500060100020100010100060500000000000101000100 0201
DATA 0002010101010100000000000000000000000000000000010203010201 0101
DATA 0101010001010001024E56FFFA396E0008EFF2302E0008907C000AC1FC0003
DATA 41FAFE8C22082041D0C041D02D48FFFA3D7C0001FFFE601E206EFFFA52AEFF
DATA FA1010322EFFFEE38141ECEFF2D0C148803080526EFFFE0C6E0004FFFE6DDA
DATA 2F2CEF704EBAFE2E588F302CEFA04E5E4E754E56FFFE41ECEFF22948EF7441
DATA ECEFD22948EF7841ECEFB02948EF7C41ECEFA02948EF8041ECEF942948EF84
DATA 41ECEF8C2948EF8841ECEF742948EF703F3C000A4EBAFF58548F396CEFA0EF
DATA 6E302CEF6E4E5E4E754E560000296E0008EF943F3C00174EBAFF36548F4E5E
```

```
DATA 4E754E560000396E0008EFB0296E000AEF943F3C00234EBAFF18548F4E5E4E
DATA 754E560000396E0008EFB0296E000AEF943F3C00344EBAFEFA548F4E5E4E75
DATA 4E560000396E0008EFB0296E000AEF943F3C004E4EBAFEDC548F4E5E4E754E
DATA 560000296E0008EF94296E000CEF983F3C005A4EBAFEBE548F206E001030AC
DATA EFA2302CEFA04E5E4E754E5600003F2E00084EBA0F22548F0C6E830000086E
DATA 06303C0000601E3F2E00083F3C003E4EBA1048588F3940E92A6706303CFFFF
DATA 6004303C00004E5E4E754E56FFFE486C01462F2E00084EBA21A6508F4A4067
DATA 60486C014C2F2E00084EBA2194508F4A40674E486C01522F2E00084EBA2182
DATA 508F4A40673C42672F2E00083F3C003C4EBA0FEA508F3D40FFFE0C4000006C
DATA 0C396EFFFEE92A303CFFFF6038302E000CC07C20003F003F2EFFFE4EBA0E32
DATA 588F601E303C0002322E000CC27C200080413F002F2E00084EBA0ECC5C8F3D
DATA 40FFFE302EFFFE4E5E4E754E5600003F2E00083F3C004C4EBA0F86588F4E5E
DATA 4E754E56FFFE2F0B41ECE9BA26486016302B000AC07C000367082F0B4EBA00
DATA 2A588FD6FC0014200B41ECE9BA2208D2BC000005B4B0816DD83F2E00084EBA
DATA FFAE548F265F4E5E4E754E5600002F0B266E00082F0B4EBA0044588F4A4067
DATA 06303CFFFF6030302B000AC07C0010670A2F2B00064EBA0D68588F426B000A
DATA 3F2B000C4EBAFE90548F4A406706303CFFFF6004303C0000265F4E5E4E754E
DATA 56000048E70110266E0008302B000AC07C00036608303CFFFF600000982E13
DATA 9EAB0006302B000AC07C00806756302B000AC07C00026606303CFFFF607630
DATA 2B000AC07C000467103F3C000242A73F2B000C4EBA0BC2508F3F072F2B0006
DATA 3F2B000C4EBA21EC508F0C40FFFF6606303CFFFF6040026BFF7F000A300748
DATA C0D1AB000E60240C6B0000000C6F1C3F3C0001302B0004444048C02F003F2B
DATA 000C4EBA0B76508F2740000E26AB0006426B0004303C00004CDF08804E5E4E
DATA 754E56FFFC2F0B266E0008302B000AC07C00016606006B0040000A302B000A
DATA C07C00606710303CFFFF6000010026AB0006426B00046052302B000AC07C00
DATA 08671E200B41ECE9BA908881FC001441ECE96ED1C041D02008268027400006
DATA 602A302B001248C02F004EBA0C08588F2680274000060C8000000000660800
DATA 6B0008000A6006006B0010000A0CAB0000000000667A42D4BFFFC3F3C0001
DATA 3F2B000C42A73F3C00424EBA0D84DEFC000A206EFFFC2140000E26AB00060C
DATA 6B8300000C660A486CE9CE4EBAFE7E588F302B000AC07C01086706303C0001
DATA 6004302B00123F002F133F2B000C4EBA1566508F37400004536B0004302B00
DATA 040C4000006C200C6BFFFF00046608006B0020000A6006006B0040000A426B
DATA 0004303CFFFF600C2053529310104880C07C00FF265F4E5E4E754E56FFFC2F
DATA 0B266E000A426B0004302B000AC07C00026606006B0040000A302B000AC07C
DATA 00406708303CFFFF6000010A6052302B000AC07C0008671E200B41ECE9BA90
DATA 8881FC001441ECE96ED1C041D020082680274000066602A302B001248C02F00
DATA 4EBA0ADC588F2680274000060C8000000000608006B0008000A6006006B00
DATA 10000A0CAB00000000000667A4302B000AC07C00086710205352931OAE0009
DATA 006B0080000A603C302B000AC07C01006732006B0080000A2053529310AE00
DATA 09102E00094880B07C000A6718201390AB0006B06B00126C0C102E00094880
DATA C07C00FF60542F0B4EBAFD2C588F4A40670C006B0040000A303CFFFF603C30
DATA 2B000AC07C01086712426B0004102E00094880C07C00FF60226020302B0012
DATA 534037400004006B0080000A102E000920535293108048O0C07C00FF265F4E
DATA 5E4E754E5600002F0B266E000A302B000AC07C008066082013B0AB00066606
DATA 303CFFFF6010526B0004102E000953932053108048802 65F4E5E4E754E5600
DATA 00206E000853680004302800040C4000006D14206E000820105290204010 10
DATA 4880C07C00FF600A2F2E00084EBAFD2A588F4E5E4E754E560000206E000A53
DATA 680004302800040C4000006D18102E0009206E000A2210529020411080 4880
DATA C07C00FF60122F2E000A102E000948803F004EBAFE165C8F4E5E4E754E5600
DATA 002002C0BCFFFFFF006604424360265343E38264FA54433002C07C01006716
DATA 3002C07C02FF670ED4BC000002006406E28A524360E2E28AE08AC67C00FFEF
DATA 43484284433010C07C80008440484220824CDF03FF4E5E4E754E5600004E56
DATA 000048E7F080206E000820182210206E000C241826104A806A084680468108
```

```
DATA C0001F4A826A084682468308C2001F928391824CDF010F4E5E40ECE9564E75
DATA 4E56000048E7FFC0206E0008226E000C2410E182E78208C2001F30280004EA
DATA 4884403610E843C67C07FF670000DE2E11E187E78708C7001F30290004EA48
DATA 8E403A11E845CA7C07FF670000C09645D67C03FE42864284303C001FE381BE
DATA 826506660AB886620652419C849587E28FE29451C8FFE824013011B1506000
DATA 01484E56000048E7FFC0206E00082410E182E78208C2001F30280004EA4884
DATA 403610E843C67C07FF6764226E000C2E11E187E78708C7001F30290004EA48
DATA 8E403A11E845CA7C07FF6744967C03FFD645280248442A0748453204C2C530
DATA 02C0C7CEC4C4C5484748424284428538073A0242474242D087D384D082D385
DATA 4A80670408C1000124013011B150600000BC4282600000B64E56000048E7FF
DATA C0226E000C303C80003940E954B151600000124E56000048E7FFC0226E000C
DATA 426CE954206E00082C10E186E78608C6001F30280004EA488C403810E844C8
DATA 7C07FF660242862E11E187E78708C7001F30290004EA488E403A11E845CA7C
DATA 07FF660242873210E28EE28F24063604300490456E146604BC876C0E38052C
DATA 073A032E023604240630914A87671C300490456708B07C001F6E10E0AF3011
DATA B3400800000F67024487D487302CE954B1512002C0BCFFFFFF006606428142
DATA 8260485343E38264FA54433002C07C010067163002C07C02FF670ED4BC0000
DATA 01006406E28A524360E2C47CFE002202E88AE08AC67C07FFE9434842C47C00
DATA 0F84433010C07C80008440484220C2E9414841424120814CDF03FF4E5E4E75
DATA 4E56FFF648E7E080302E0008C07CFF003D40FFFE026E00FF00080C6E2000FF
DATA FE6620206E000E301048C02D40FFF6486EFFF64EBA0116588F41EEFFF62D48
DATA 000E603E0C6E2800FFFE67080C6E1000FFFE662E206E000E2D50FFF60C6E28
DATA 00FFFE660C486EFFF64EBA00E2588F600A486EFFF64EBA009C588F41EEFFF6
DATA 2D48000E2F2E000E2F2E000A302E0008E58041ECE930D0C020504E90508F4C
DATA DF01074E5E2F57000ADEFC000A44ECE9564E754E5E4E752940E94C486CE94C
DATA 4EBA008E584F4E75486CE94C4EBA000A584F202CE94C4E754E56000048E7C0
DATA 80206E000820102228000466044A80671CE381E390E381E390E381E3902210
DATA C2BC80000000D0BC400000000808120804CDF01034E5E4E754E56000048E7C0
DATA 80206E000842812010660442806018E280E291E280E291E280E291C0BC8FFF
DATA FFFFD0BC3800000020C020814CDF01034E5E4E754E56000048E7FFC0206E00
DATA 0824106606424342826018 6E024482363C041D2002C0BCFFFFFF006606E182
DATA 967C00084EFAFE1C4E5E4E754E56FFF248E70730266E0008246E000C426EFF
DATA FA2D4AFFFC200B20402950E94C29680004E9502D6CE94CFFF22D6CE950FFF6
DATA 424547EEFFF23E13E84FCE7C07FF670000809E7C03FF4A476E08BE7CFFFD6D
DATA 1260205245486CE9662F0B4EBAFBF4504F60D45345486CE9662F0B4EBAFC62
DATA 504F60C42613E183E783302B0004EA48864008C3001F4A4767044447EEAB30
DATA 3C0030E3836402524014C0302E00144241E383E3512C033E01E383E351E383
DATA E351D686D347D27C003014C151C8FFE242126004526EFFFA302EFFFA671E24
DATA 6EFFFC4287600614FC003052872007322E001448C1B0816DEE42126062202E
DATA FFFC362E001448C3D08352802440206EFFFC10104880B07C00306602528A5A
DATA 2260060412000A522210124880B07C00396EF0206EFFFC10104880B07C0030
DATA 6610246EFFFC200A5280264014DB66FC5345202EFFFC362E001448C3D08320
DATA 404210206E001030854CDF0CE04E5E4E754E56FFFC48E70130266E000C426E
DATA FFFC10134880B07C006266083D7C2000FFFC528B10134880B07C0072671E10
DATA 134880B07C0077671410134880B07C0061670A203C000000060000146202C
DATA E92C24400C8000000000662A41ECE9BA24486010302A000AC07C0003660260
DATA 16D4FC0014200A41ECE9BA2208D2BC000005B4B0816DDE200A41ECE9BA2208
DATA D2BC000005B4B0816D0A203C00000000600000F242ACE92C426A000A426EFF
DATA FE102B00014880B07C002B660C3D7C0002FFFE006A0003000A10134880B07C
DATA 007766183F2EFFFC2F2E00084EBAF45C5C8F3E00006A0002000A6068101348
DATA 80B07C00616642303C0002806EFFFC3F002F2E00084EBA038C5C8F3E000C40
DATA FFFF66103F2EFFFC2F2E00084EBAF41E5C8F3E003F3C000242A73F074EBA01
DATA 6E508F006A0006000A601C302EFFFE806EFFFC3F002F2E00084EBA034A5C8F
```

```
DATA 3E00006A0001000A0C47FFFF660C426A000A203C00000000603035470000C42
DATA 6A0004203C00000000248025400006 3F3C000142A73F2A000C4EBA0114508F
DATA 2540000E356CE9B80012200A4CDF0C804E5E4E754E56FFFE48E70030266E00
DATA 08244B6010302EFFFE14C04880B07C000A66026048536E000C302E000C0C40
DATA 00006F3A206E000E536800043028000 40C4000006D14206E000E2010529020
DATA 4010104880C07C00FF600A2F2E000E4EBAF548588F3D40FFFE0C40FFFF66A8
DATA 42120C6EFFFFFFFE660E200AB08B6608203C000000006002200B4CDF0C004E
DATA 5E4E754E56FFFE48E7F000242E0008660A80FC000042804281604A222E000C
DATA 426EFFFE4A816C06566EFFFE44814A826C06526EFFFE44827601B4816406E3
DATA 82E38360F64280B481620480839282E28AE28B64F20C6E0003FFFE6D024481
DATA E2EEFFFE640244802D4000082D41000C4CDF000F4E5E4E754E56FFF20C6E00
DATA 0000086C0A203CFFFFFFFF600000FE3F2E000E3F2E00082F2E000A3F3C0042
DATA 4EBA02A8DEFC000A2D40FFFC0C80000000006D08202EFFFC600000D23F3C00
DATA 013F2E000842A73F3C00424EBA027EDEFC000A2D40FFF83F3C00023F2E0008
DATA 42A73F3C00424EBA0264DEFC000A2D40FFF40C6E0001000E660E202EFFF8D0
DATA AE000A2D40000A60240C6E0002000E660E202EFFF4D0AE000A2D40000A600E
DATA 302E000E6708203CFFFFFFFF6062202E000AB0AEFFF46F1E486EFFF2202E00
DATA 0A90AEFFF42F003F2E00083F3C00404EBA01FEDEFC000C42673F2E00082F2E
DATA 000A3F3C00424EBA01E8DEFC000A0C80000000005DC0C07C00013940E92A67
DATA 0A203CFFFFFFFF600A6008426CE92A202E000A4E5E4E754E5600002F2E0008
DATA 3F3C00484EBA01AC5C8F4E5E4E754E5600002F2E00083F3C00494EBA01965C
DATA 8F4E5E4E754E56FFFE426EFFFE6036302EFFFEE58041ECE7FAD0C030106622
DATA 302EFFFEE58041ECE7FAD0C030AE0008302EFFFEE58041ECE7FCD0C030AE00
DATA 0A600C526EFFFE0C6E004CFFFE6DC24E5E4E754E56FFFE426EFFFE6026302E
DATA FFFEE58041ECE7FAD0C03010B06E0008660E302EFFFEE58041ECE7FAD0C042
DATA 50526EFFFE0C6E004CFFFE6DD24E5E4E754E56FFFE3F073E2E00083D7C0130
DATA FFFE41ECE7FA2248D2EEFFFEBE5067065848B3C866F6302800023E1F4E5E4E
DATA 754E56FFFE486C016C2F2E00084EBA124E508F4A40660A3D7C8300FFFE6000
DATA 009E486C01722F2E00084EBA1232508F4A40660A3D7C82FFFFFE6000008248
DATA 6C01782F2E00084EBA1216508F4A4066083D7C82FEFFFE6066302E000CC07C
DATA 000167302F2E00084EBA130E588F4A406706303CFFFF606042672F2E00083F
DATA 3C003C4EBA0058508F3F003F3C003E4EBA004C588F302E000CC07C00033F00
DATA 2F2E00083F3C003D4EBA0034508F3D40FFFE3940E92A0C4000006C06303CFF
DATA FF6018302E000CC07C20003F003F2EFFFE4EBAFE7E588F302EFFFE4E5E4E75
DATA 2949E7F6294AE7F2295FE7EE4E41226CE7F6246CE7F22F2CE7EE4E754E56FF
DATA D648E70110266E000C426EFFD6302E0008B07C006467026008 3E3C000A602E
DATA 6008B07C007567026008 3E3C000A601E6008B07C006F670260083E3C000860
DATA 0E6008B07C007867026004 3E3C00100CAE0000000000106C2A0C6E00640008
DATA 66142053529310BC002D202E001044802D400010600E302E000A660802AE00
DATA 00FFFF00100C470008660C202E0010C0BC0000000760260C470010660C202E
DATA 0010C0BC0000000F6014202E00102F00300748C02F004EBAFBEC588F201F32
DATA 2EFFD6526EFFD6E38141EEFFD8D0C130800C470008660E202E0010E680C0BC
DATA 1FFFFFFF60280C470010660E202E0010E880C0BC0FFFFFFF6014202E00102F
DATA 00300748C02F004EBAFB9E201F588F2D400010202E00106600FF706046536E
DATA FFD6302EFFD6E38041EEFFD8D0C00C50000A6D18302EFFD6E38041EEFFD8D0
DATA C03010907C000AD07C00416012302EFFD6E38041EEFFD8D0C03010D07C0030
DATA 205352931080302EFFD666B44CDF08804E5E4E754E56FFDE48E70310266E00
DATA 0C0C6E0100000A66063D7C0006000A296E0010E94C296E0014E950486C017E
DATA 486CE94C3F3C08044EBAF6085DC0C07C00011D40FFDE302E000A52403F0048
DATA 6EFFE0486EFFE2486E00104EBAF766DEFC000E102EFFDE670820535293 10BC
DATA 002D102EFFE24880B07C00306612102EFFE3660C2053529310AEFFE2600001
DATA 4A0C6E00660008660000E03E2EFFE0DE6E000A52473007B06CE9586F06302C
DATA E958600230073E000C4700006C06303C0001600230073F00486EFFE0486EFF
```

```
DATA E2486E00104EBAF6F0DEFC000E4246601830065246 41EEFFE2D0C010102053
DATA 529310805347536EFFE0300767080C6E0000FFE06CDC600C2053529310BC00
DATA 30536EFFE00C6E0000FFE06CEC302E000A67082053529310BC002E60102053
DATA 529310BC0030526EFFE0536E000A302E000A67080C6EFFFFFFE06DE260220C
DATA 4700006F103006524641EEFFE2D0C0101048806004303C0030205352931080
DATA 5347302E000A536E000A4A4066D260602053529310AEFFE2302E000A670820
DATA 53529310BC002E3E3C000160123007524741EEFFE2D0C01010205352931080
DATA 302E000A536E000A4A4066E22053529310BC00453F2EFFE0486C01862F134E
DATA BA033ADEFC000A2F134EBA0EDC588F48C0D1934CDF08C04E5E4E754E56FFFE
DATA 426EFFFE6022302EFFFEC1FC000A206E000820501610 4883D043907C00303D
DATA 40FFFE206E00085290206E000820501010 4880B07C00306D10206E00082050
DATA 10104880B07C00396FBE302EFFFE4E5E4E754E56FFE648E70330266E000C2D
DATA 6E0008FFFC200B20402450588B60000252600A206E000852AE0008109A1012
DATA 670A10124880B07C002566E810124880B07C00256600022C528A426EFFEC3D
DATA 7C0100FFE8426EFFE61D7C0020FFEF10124880B07C002D66083D7C0001FFEC
DATA 528A10124880B07C003066081D7C0030FFEF528A2D4AFFF0486EFFF04EBAFF
DATA 24588F3D40FFEA246EFFF010124880B07C002E6618528A2D4AFFF0486EFFF0
DATA 4EBAFF02588F3D40FFE8246EFFF010124880B07C006C66083D7C0001FFE652
DATA 8A101267041E1A48872D6E0008FFF80C47006467120C47006F670C0C470078
DATA 67060C4700756642302EFFE66708200B20402010600 8200B2040301048C02F
DATA 00486E00083F2EFFE63F074EBAFB56DEFC000C302EFFE66706303C00046004
DATA 303C000248C0D7C0600000A40C4700636618200B5280204010102 06E000852
DATA AE00081080548B600000860C4700736634200B20402D50FFF4588B3C2EFFE8
DATA 6016206EFFF452AEFFF41010206E000852AE000810805346300667 08206EFF
DATA F4101066DE604A0C470065670C0C47006667060C470067662C200B20402950
DATA E94C29680004E9502F2CE9502F2CE94C486E00083F2EFFE83F074EBAFC18DE
DATA FC0010508B600C3007206E000852AE00081080202E000890AEFFF8B06EFFEA
DATA 6C00008E302EFFEC67243C2EFFEA222E000892AEFFF89C41600E206E000852
DATA AE000810AEFFEF5346300666EE6062202EFFF8322EFFEA534148C1D0812D40
DATA FFF42D40FFF02C2E00089CAEFFF8601C202EFFF83206534148C1D081204010
DATA 10206EFFF453AEFFF41080534630066 6E0600C206EFFF852AEFFF810AEFFEF
DATA 202EFFF8B0AEFFF46FEA202EFFF052802D40000810126600FDAC206E000842
DATA 10202E000890AEFFFC4CDF0CC04E5E4E754E56FEFE2F2E000C486EFF004EBA
DATA FD68508F3D40FEFE2F2E0008486EFF004EBA0038508F302EFEFE4E5E4E754E
DATA 560000486E000C2F2E00084EBAFFC6508F4E5E4E754E560000486E000C2F2E
DATA 00084EBAFD26508F4E5E4E754E56000060182F2E000C206E000852AE000810
DATA 1048803F004EBAEE045C8F206E0008101066E04E5E4E754E56FFFC0C6E8300
DATA 00086600013A302CE79CB06CE79A66000100303C00003940E79A3940E79C60
DATA 0000E83F3C00074EBAF910548F3D40FFFC0C6E0008FFFC662A302CE79C6720
DATA 536CE79C3F3C001B3F3C00024EBAF8EC588F3F3C00443F3C00024EBAF8DE58
DATA 8F600000A80C6E000DFFFC6630302CE79C526CE79C41ECE79ED0C010BC000A
DATA 3F3C000D3F3C00024EBAF8B2588F3F3C000A3F3C00024EBAF8A4588F60760C
DATA 6E0003FFFC660A3F3C00024EBAE912548F0C6E001AFFFC6632302EFFFC322C
DATA E79C526CE79C41ECE79ED0C110803F3C000D3F3C00024EBAF866588F3F3C00
DATA 0A3F3C00024EBAF858588F602A302EFFFC322CE79C526CE79C41ECE79ED0C1
DATA 10803F2EFFFC3F3C00024EBAF834588F303C00016600FF1441ECE79ED0ECE7
DATA 9A10104880B07C001A6606303CFFFF6012302CE79A526CE79A41ECE79ED0C0
DATA 10104880601C601A0C6E82FF0008660E3F3C00034EBAF7EC548F6006600430
DATA 3CFFFF4E5E4E754E56FFFC48E70030266E000A302E000EC0BC0000FFFF2D40
DATA FFFC426CE92A0C6E830000086E4842AEFFFC601E3F2E00084EBAFE6A548F16
DATA 804880B07CFFFF660460146006528B52AEFFFC302E000E536E000E4A4066D6
DATA 202EFFFC6608303CFFFF600000E2202EFFFC600000DA602E2F2E000A2F2EFF
DATA FC3F2E00083F3C003F4EBAF75CDEFC000C2D40FFFC3940E92A0C4000006C08
```

```
DATA 303CFFFF600000AA3F2E00084EBAF634548F4A4066000092202EFFFC3D4000
DATA 0E202E000A26402440606C200B90AE000A48C0B0AEFFFC6D4A2F0A302E000E
DATA C0BC0000FFFF2F003F2E00083F3C003F4EBAF6F8DEFC000C3940E92A0C4000
DATA 006C06303CFFFF604A302CE92A6602602E302E000E906CE92AC0BC0000FFFF
DATA 91AEFFFC264A10134880B07C000D6604528B600614DB536E000E302E000E66
DATA 8E200A90AE000A48C02D40FFFC426CE92A202EFFFC4CDF0C004E5E4E754E56
DATA 0000302CE790670C2F2CE7964EBAEAFC588F600C206CE79252ACE792101048
DATA 804E5E4E754E56FFFE4EBAFFD43D40FFFE0C6E0020FFFE67F00C6E0009FFFE
DATA 67E80C6E000AFFFE67E0302EFFFE4E5E4E754E56FFFE3D7CFFFFFFFE0C6E00
DATA 3000086D160C6E003900086E0E302E0008907C00303D40FFFE60320C6E0061
DATA 00086D12302E0008907C0061D07C000A3D40FFFE60180C6E004100086D1030
DATA 2E0008907C0041D07C000A3D40FFFE302EFFFEB06E000A6F063D7CFFFFFFFE
DATA 0C6EFFFFFFFE6606397C0001E78E302EFFFE4E5E4E754E56FFF62F073D6E00
DATA 0AFFFC426EFFF8426EFFF64287302E0008B07C00646702600A3D7C000AFFFE
DATA 60346008B07C006F6702600A3D7C0008FFFE60226008B07C00786702600A3D
DATA 7C0010FFFE60106008B07C006867026006 3D7C000AFFFE0C6E002DFFFC6610
DATA 303C00013D40FFF84EBAFEBE3D40FFFC60560C6E000AFFFE66162007E78022
DATA 07E381D081362EFFFA48C3D0832E0060260C6E0008FFFE66102007E780362E
DATA FFFA48C3D0832E00600E2007E980362EFFFA48C3D0832E00536E000C3D7C00
DATA 01FFF64EBAFE663D40FFFC302E000C67183F2EFFFE3F2EFFFC4EBAFEA2588F
DATA 3D40FFFA0C40FFFF668C302EFFF6670A302CE78E6704426CE78E302CE79067
DATA 102F2CE7963F2EFFFC4EBAE8F65C8F600453ACE792302EFFF8670620074480
DATA 600220072E1F4E5E4E754E56FFDC48E701103E2E00083D7C000AFFFE302EFF
DATA FE48C04EBAECF02D6CE94CFFF62D6CE950FFFA426EFFF0426EFFEE426EFFEC
DATA 426EFFEA3D7C0001FFE8426EFFE63D7C0013FFE4296C018AE94C296C018EE9
DATA 502D6CE94CFFDC2D6CE950FFE00C47002D660E1D7C0001FFF54EBAFD963E00
DATA 6004422EFFF5426EFFF2600000F80C47002D6610302EFFEE670A3D7C0001FF
DATA EA600000D20C4700306D00009A0C4700396E000092302EFFEE6714302EFFEC
DATA C1FC000AD047907C00303D40FFEC60740C4700306704426EFFE8302EFFE466
DATA 06526EFFF2605E302EFFF06704536EFFF2302EFFE8664E296EFFDCE94C296E
DATA FFE0E950486EFFF6486CE94C3F3C08024EBAEB583F072F0F486CE94C3F3C20
DATA 004EBAEB48548F486C0192486CE94C3F3C08014EBAEB362D6CE94CFFDC2D6C
DATA E950FFE0536EFFE460300C47002E660E302EFFF066083D7C0001FFF0601C0C
DATA 47004567060C470065660E302EFFEE66083D7C0001FFEE600260183D7C0001
DATA FFE64EBAFC963E00536E000A302E000A6600FF04302EFFE66606397C0001E7
DATA 8E302EFFEA670A302EFFEC916EFFF26008302EFFECD16EFFF260620C6E0000
DATA FFF26C2E296EFFDCE94C296EFFE0E950486EFFF6486CE94C3F3C08034EBAEA
DATA 922D6CE94CFFDC2D6CE950FFE0526EFFF2602C296EFFDCE94C296EFFE0E950
DATA 486EFFF6486CE94C3F3C08024EBAEA642D6CE94CFFDC2D6CE950FFE0536EFF
DATA F2302EFFF26698102EFFF56728296EFFDCE94C296EFFE0E950486C019A486C
DATA E94C3F3C08024EBAEA2C2D6CE94CFFDC2D6CE950FFE0302CE790670E2F2CE7
DATA 963F074EBAE6905C8F600453ACE792296EFFDCE94C296EFFE0E9504CDF0880
DATA 4E5E4E754E56FFE248E70030426EFFFE426CE78E206E000858AE0008265060
DATA 00032E101B48803D40FFF80C6E0020FFF867100C6E0009FFF867080C6E000A
DATA FFF8662E4EBAFB843D40FFF60C6EFFFFFFF6671A302CE79067102F2CE7963F
DATA 2EFFF64EBAE6145C8F600453ACE792600002E00C6E0025FFF867484EBAFB4E
DATA 3D40FFF60C6EFFFFFFF66608303CFFFF600002CA302EFFF6B06EFFF8672230
DATA 2CE79067102F2CE7963F2EFFF64EBAE5CC5C8F600453ACE792302EFFFE6000
DATA 029E60000290303C00003D40FFF23D40FFF03D7C0100FFEE10134880B07C00
DATA 2A66083D7C0001FFF2528B10134880B07C00306D2010134880B07C00396E16
DATA 2D4BFFFA486EFFFA4EBAF490588F3D40FFEE266EFFFA10134880B07C006C66
DATA 083D7C0001FFF0528B101B48803D40FFF40C6E0063FFF466344EBAFA703D40
DATA FFF60C40FFFF6608303CFFFF60000214302EFFF26610302EFFF6206E000858
```

```
DATA AE000820501080526EFFFE600001EC0C6E0064FFF4671A0C6E006FFFF46712
DATA 0C6E0078FFF4670A0C6E0068FFF4660000904EBAFA403D40FFF60C40FFFF66
DATA 08303CFFFF600001BE3F2EFFEE3F2EFFF63F2EFFF44EBAFACA5C8F2D40FFE2
DATA 302CE78E6708302EFFFE6000019A302EFFF26646302EFFF06712206E000858
DATA AE00082010204020AEFFE2602E0C6E0068FFF46614202EFFE2206E000858AE
DATA 000822102041108060122O2EFFE2206E000858AE000822102041308052 6EFF
DATA FE6000013C0C6E0065FFF4670A0C6E0066FFF4660000964EBAF9A03D40FFF6
DATA 0C40FFFF6608303CFFFF6000011E3F2EFFEE3F2EFFF64EBAFB58588F2D6CE9
DATA 4CFFE62D6CE950FFEA302CE78E6708302EFFFE600000F6302EFFF26648302E
DATA FFF06724296EFFE6E94C296EFFEAE950206E000858AE00082210204120ACE9
DATA 4C216CE9500004601E296EFFE6E94C296EFFEAE9504EBAE816206E000858AE
DATA 000822102041208052 6EFFFE600000960C6E0073FFF46600008C4EBAF9023D
DATA 40FFF60C40FFFF6608303CFFFF60000080302EFFF2660A206E000858AE0008
DATA 24506024302EFFF26606302EFFF614C04EBAF8A83D40FFF60C40FFFF660630
DATA 3CFFFF604C536EFFEE302EFFEE67180C6E0020FFF667100C6E0009FFF66708
DATA 0C6E000AFFF666BE421A302CE79067102F2CE7963F2EFFF64EBAE3365C8F60
DATA 0453ACE792526EFFFE10136600FCD0302EFFFE4CDF0C004E5E4E754E560000
DATA 296E0008E792426CE790486E000C4EBAFC90588F4E5E4E754E56FFFC48E700
DATA 30266E0008246E000C2D4BFFFC60000002101B6600FFFC538B6000000216DA
DATA 6600FFFC202EFFFC600000024CDF0C004E5E4E754E56000048E70030266E00
DATA 08246E000C60000006528B528A10136700001010131212488148 80B0416700
DATA FFEA10134880161248839043600000024CDF0C004E5E4E754E56000048E701
DATA 30266E0008246E000C3E2E00106000000 6528B528A53473007670000161013
DATA 67000010101312124881488 0B0416700FFE2101348801612488390436000000
DATA 024CDF0C804E5E4E754E560000206E00082008226E000C10D96600FFFC4E5E
DATA 4E754E56000048E70030266E0008244B60000004528B10136600FFFA200B90
DATA 8A600000024CDF0C004E5E4E754E56000048E70310266E00083E2E000C6000
DATA 0004528B10131C006700000C10064880B0476600FFEE10066700000 8200B60
DATA 000008203C000000006000000 24CDF08C04E5E4E754E5600002F2E00083F3C
DATA 00414EBAED565C8F3940E92A6708303CFFFF60066004303C00004E5E4E754E
DATA 5600006034206E000810104880B07C000A660E3F3C000D3F3C00024EBAED1E
DATA 588F206E000852AE0008101048803F003F3C00024EBAED06588F302E000C53
DATA 6E000C4A4066C04E5E4E754E5600006018206E000852AE0008101048803F00
DATA 3F3C00044EBAECD8588F302E000C536E000C4A4066DC4E5E4E754E56000060
DATA 18206E000852AE0008101048803F003F3C00054EBAECAA588F302E000C536E
DATA 000C4A4066DC4E5E4E754E56FFF248E70030266E000A244B41EC01A22D48FF
DATA F60C6E8300000866163F2E000E2F0B4EBAFF345C8F3D6E000EFFFE6000014C
DATA 0C6E82FF000866163F2E000E2F0B4EBAFF605C8F3D6E000EFFFE6000012E0C
DATA 6E82FE000866163F2E000E2F0B4EBAFF705C8F3D6E000EFFFE600001102D4B
DATA FFFA426EFFFE3F2E00084EBAEB12548F4A40660000A6600000941012 4880B0
DATA 7C000A66000086200A90AEFFFAB07C00006F40200A90AEFFFA48C02D40FFF2
DATA 2F2EFFFA2F2EFFF23F2E00083F3C00404EBAEBD4DEFC000C3940E92A48C0B0
DATA AEFFF26708303CFFFF600000A8302CE92AD16EFFFE2F2EFFF62F3C00000002
DATA 3F2E00083F3C00404EBAEB9EDEFC000C3940E92A0C4000026706303CFFFF60
DATA 74526EFFFE528A200A2D40FFFA6002528A200A908BB06E000E6500FF646010
DATA 200B362E000EC6BC0000FFFFD0832440200A90AEFFFA48C02D40FFF22F2EFF
DATA FA2F2EFFF23F2E00083F3C00404EBAEB3CDEFC000C3940E92A48C0B0AEFFF2
DATA 6706303CFFFF6010302CE92AD16EFFFE426CE92A302EFFFE4CDF0C004E5E4E
DATA 7543ECE9BA22FC0000000032FC000022FC0000000032FC000932FC830022FC
DATA 0000000032FC000022FC0000000032FC000022FC0000000032FC010232FC83
DATA 0022FC0000000032FC020022FC0000000032FC000022FC0000000032FC0102
DATA 32FC830022FC0000000032FC0200D2FC0578397C0200E9B8296C0158E96629
DATA 6C015CE96A296C0160E962296C0164E95A296C0168E95E397C0007E95843EC
```

```
DATA E93041ED000622C841ED001E22C841ED001822C841ED002422C841ED002A22
DATA C841ED001222C841ED000C22C8297C00000000E92C397C0000E79C397C0000
DATA E79A4E7520205555434F444500005B325D5B5555434F44457C576869636820
DATA 6F7065726174696F6E3F5D5B456E636F64657C4465636F64657C43616E6365
DATA 6C5D00003A5C2A2E2A00627200005B335D5B46696C65206E6F7420666F756E
DATA 642E5D5B43616E63656C5D00002E555545000077005B335D5B43616E277420
DATA 6F70656E202E5555452066696C655D5B43616E63656C5D00626567696E2036
DATA 34342025730A00656E640A0000750A000072005B335D5B43616E2774206669
DATA 6E64202E5555452066696C655D5B43616E63656C5D00626567696E20000025
DATA 7320257320257300006277000 05B335D5B43616E2774206F70656E206F7574
DATA 7075747C66696C655D5B43616E63656C5D00656E640A00005B315D5B4E6F20
DATA 656E64206C696E652E5D5B436F6E74696E75655D005B335D5B53686F727420
DATA 66696C65215D5B436F6E74696E75655D00002000434F4E3A00004155583A00
DATA 005052543A00004024000000000013F8000003FF000000000000434F4E3A
DATA 00004155583A00005052543A0000000000000000000025640000000000000000
DATA 0000004048000000000000BFF00000000000000D0A0000000000206060606
DATA 0606060600
'-------------------------CUT HERE-------------------------------


From -5739967978035731473
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,611120cd427f17f2,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1990-06-01 22:02:22 PST
Path: gmdzi!unido!mcsun!uunet!munnari.oz.au!metro!sunaus.oz!newstop!male!
panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i061: chaos -- Draw random Sierpinski Triangles
Keywords: ST-BASIC
Message-ID: <2116@male.EBay.Sun.COM>
Date: 2 Jun 90 01:37:14 GMT
Sender: news@male.EBay.Sun.COM
Reply-To: WJOHNSTON@COLGATEU.BITNET (Michael Johnston)
Lines: 54
Approved: koreth@panarthea.ebay.sun.com
Posted: Sat Jun  2 02:37:14 1990

Submitted-by: WJOHNSTON@COLGATEU.BITNET (Michael Johnston)
Posting-number: Volume 3, Issue 61
Archive-name: chaos

10    'This program, written in ST Basic, draws the "Sierpinski
20    'Triangle" in randomly-selected sizes and shapes. It illustrates
30    'some principles of chaos theory, such as fractals and self-
40    'similarity, bounded randomness, and constant complexity
50    'across orders of scale. A discussion of the "affine transfor-
60    'mations" used to create this diagram was presented on the PBS
70    'program "Nova" in January 1989, and also in the May 1990 issue
80    'of SCIENTIFIC AMERICAN. Have fun with this one--
90    '        Michael Johnston  (WJOHNSTON@COLGATEU.BITNET)
100   FULLW 2:CLEARW 2:CT=0
```

```
110   'You can change colors by playing with line 130; green on
120   'black is nice, for example.
130   COLOR 1,1,1
140   FOR C=1 TO 3
150   RANDOMIZE 0
160   XC(C)=RND(0)
170   XC(C)=XC(C)*1000
180   IF XC(C)>606 THEN 150
190   NEXT
200   FOR C=1 TO 3
210   RANDOMIZE 0
220   YC(C)=RND(0)
230   YC(C)=YC(C)*1000
240   IF YC(C)>166 THEN 210
250   NEXT
260   PCIRCLE XC(1), YC(1), 1
270   PCIRCLE XC(2), YC(2), 1
280   PCIRCLE XC(3), YC(3), 1
290   RANDOMIZE 0
300   X=RND(0)
310   X=X*1000
320   IF X>606 THEN 290
330   RANDOMIZE 0
340   Y=RND(0)
350   Y=Y*1000
360   IF Y>166 THEN 330
370   GOTO 440
380   XCH=(X-XC(1))/2:YCH=(Y-YC(1))/2:GOTO 410
390   XCH=(X-XC(2))/2:YCH=(Y-YC(2))/2:GOTO 410
400   XCH=(X-XC(3))/2:YCH=(Y-YC(3))/2
410   NX=X-XCH:NY=Y-YCH:IF CT < 15 THEN 430
420   PCIRCLE  NX, NY, 0
430   X=NX:Y=NY
440   RANDOMIZE 0:CT=CT+1
450   D=RND(0)*10
460   IF D=6 OR D>6 THEN 440
470   D=INT(D)
480   IF D=0 OR D=1 THEN 380
490   IF D=2 OR D=3 THEN 390
500   IF D=4 OR D=5 THEN 400
```


From 499592218625499628
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,4cc0e0beb512f677,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1990-06-01 22:25:01 PST
Path: gmdzi!unido!mcsun!sunic!uupsi!rpi!zaphod.mps.ohio-state.edu!uwm.edu!lll-winken!sun-barr!newstop!male!panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i062: bifurc -- Draw bifurcation diagrams
Keywords: ST-BASIC

Message-ID: <2117@male.EBay.Sun.COM>
Date: 2 Jun 90 01:38:49 GMT
Sender: news@male.EBay.Sun.COM
Reply-To: WJOHNSTON@COLGATEU.BITNET (Michael Johnston)
Lines: 49
Approved: koreth@panarthea.ebay.sun.com
Posted: Sat Jun 2 02:38:49 1990

Submitted-by: WJOHNSTON@COLGATEU.BITNET (Michael Johnston)
Posting-number: Volume 3, Issue 62
Archive-name: bifurc

```
100   'This program, written in ST Basic, uses a "difference equation"
110   '(line 330) to model the effects of driving a dynamic system at
120   'increasingly high rates. The result is a bifurcation diagram
130   'which represents the growth rates of biological populations
140   '(on the Y axis) at increasingly high rates of fertility (X axis). At
150   'low to moderate fertility rates, rates of increase
160   'reach an equilibrium. But at higher rates, a boom-and-bust
170   'cycle is revealed in the form of "period doubling"; and further
180   'to the right, the results are chaotic. For a more detailed
190   'discussion, see James Gleick, CHAOS: MAKING A NEW SCIENCE
200   '(Viking 1987), pp. 59-80. Be forewarned: this program takes
210   ' a LOOOONNNNNGGGGGGG time to run ( ~ 40 minutes), but the
220   'results are interesting. Note also that by playing with the
230   'values of R and X (line 310), and the integer values in
240   'the second statements of lines 390 and 510, you can
250   '"zoom in" on various segments of this diagram.
260   'Have fun--
270   '          Michael Johnston  (WJOHNSTON@COLGATEU.BITNET)
280   COLOR 1,1,1
290   FULLW 2:CLEARW 2
300   DIM REG (167)
310   R=1.01:X=.02:ST=0:P=0:T=80:VS=0
320   ST=ST+1
330   XN=R*X*(1-X)
340   IF ABS(XN-X) < .000001 THEN 380
350   IF ST>T THEN 420
360   X=XN
370   GOTO 320
380   R=R+0.01:X=.02
390   V=166-(XN*166):H=((R-1)/4)*600
400   PCIRCLE H,V,0
410   ST=0:GOTO 320
420   P=P+1:VS=VS+1:IF VS>1750 THEN T=40
430   REG(P)=XN
440   IF P<3 THEN GOTO 360
450   FOR I=1 TO P-1
460   IF ABS(REG(P)-REG(I))<.000001 THEN GOTO 500
470   NEXT
480   IF P=166 THEN GOTO 500
490   GOTO 360
500   FOR I=1 TO P-1
```

```
510   V=166-(REG(I)*166):H=((R-1)/4)*600
520   PCIRCLE H,V,0
530   NEXT
540   ST=0:P=0:R=R+0.01:X=.02:GOTO 320
```

```
From -7621313600524266490
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,42ed967d6e44194e,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1990-08-14 13:57:37 PST
Path: gmdzi!unido!mcsun!uunet!cs.utexas.edu!sun-barr!newstop!male!
panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i087: sozo1.2pat -- Patches for Sozobon C 1.2
Keywords: text
Message-ID: <2954@male.EBay.Sun.COM>
Date: 14 Aug 90 20:57:37 GMT
Sender: news@male.EBay.Sun.COM
Reply-To: nike.cair.du.edu!onecom!wldrdg!hans (Johann Ruegg)
Lines: 58
Approved: koreth@panarthea.ebay.sun.com
Posted: Tue Aug 14 21:57:37 1990

Submitted-by: nike.cair.du.edu!onecom!wldrdg!hans (Johann Ruegg)
Posting-number: Volume 3, Issue 87
Archive-name: sozo1.2pat

Here is a bug fix for "top" in Sozobon release 1.2.  You will need it if
you use the -O option at all.

Basically, several calls to the function "uses" or "sets" were incorrectly
coded with register masks instead of register numbers.  The fix is to
take out the RM() calls in several places.

Also, line 299 of peep3.c should call "uses" instead of "sets" with the
first parameter being "i2" not "i3".


In "peep2.c", make the following changes:

688c688
<             if (uses(i2, RM(dr1)))
---
>             if (uses(i2, dr1))
736c736
<             if (uses(i2, RM(dr1)))
---
>             if (uses(i2, dr1))
784c784
<             if (uses(i2, RM(dr1)))
---
```

```
>             if (uses(i2, dr1))
```

In "peep3.c", make the following changes:

```
299c299
<        !sets(i3, RM(i3->src.areg))) {
---
>        !uses(i2, i3->src.areg)) {
333c333
<                 if (sets(i3, RM(rm)))
---
>                 if (sets(i3, rm))
400c400
<                 if (sets(i3, RM(rm)))
---
>                 if (sets(i3, rm))
467c467
<                 if (sets(i3, RM(rm)))
---
>                 if (sets(i3, rm))
```

After you have made these changes, recompile the optimizer "top" without the
-O option (dont use the old "top"). Then install this fixed version of "top".
Now recompile again with the -O option (e.g. using the newly make "top").
Now install this "top" as your new version.

     Johann Ruegg
     Sozobon Ltd.
     uunet!isis!onecom!wldrdg!hans

From -2973188066917529680
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,598ec37e630c9ad8,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1990-08-14 14:05:05 PST
Path: gmdzi!unido!mcsun!uunet!wuarchive!cs.utexas.edu!sun-barr!newstop!male!
panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i090: nroff110pat -- Patch for Nroff 1.10
Keywords: diff
Message-ID: <2959@male.EBay.Sun.COM>
Date: 14 Aug 90 21:05:05 GMT
Sender: news@male.EBay.Sun.COM
Reply-To: rosenkra%c1yankee@convex.com (Bill Rosenkranz)
Lines: 176
Approved: koreth@panarthea.ebay.sun.com
Posted: Tue Aug 14 22:05:05 1990

Submitted-by: rosenkra%c1yankee@convex.com (Bill Rosenkranz)
Posting-number: Volume 3, Issue 90

Archive-name: nroff110pat

This is patch 1 for nroff v1.10 for atari/TOS (diff -c files below, no shar, they are small).

Note: there is a GNU version of troff (groff). Most likely it will not fit under minix, but it should work under TOS. It will probably need gcc to compile, however. I have not tried this. It comes with tbl, eqn, and pic, I believe.

Note: if you wish to compile nroff under TOS, you will need ctype (isalpha, isspace, islower, isdigit, isprint), stdio (of course), strings (strncmp, strcmp, strlen, strcpy, strcat, atoi, tolower), abs (macro normally), and unix compatible versions of getenv(3), time(3), and ctime(3). Most compilers support these, or you could get the PD/sharewhare (???) dLibs package which is very good. For unix, you will also need tgetstr(3) from termcap (or terminfo). I use Alcyon 4.14 with my own libc.a. There is also a single call to GEMDOS Cconin (to support the -h option).

Note: if you feel up to making changes to this nroff, pls mail the changes to me in "diff -c" format so I can try and keep this version up to date.

This patch adds the "-raN" switch to the command line to preset a (single char) number register (a) to decimal integer value N. In this version of nroff, there are only single char (user) number registers and case is ignored (thus there are 26 number registers besides the preset ones).

To test, the following file (I called it "nr.nr"):

```
% cat nr.nr
.\" file: nr.nr
.\" test "nroff -raN ..." (set number register)
.pl 20
register a = \na
.br
register b = \nb
.br
register c = \nc
.br
register d = \nd
.br
register e = \ne
.br
register f = \nf
.br
register g = \ng
.br
register h = \nh
.br
register i = \ni
.br
register j = \nj
.br
```

when used as follows:

```
% nroff -rc10 -rg20 nr.nr >xxx
```

produces the following output (file xxx):

```
% cat xxx




register a = 0
register b = 0
register c = 10
register d = 0
register e = 0
register f = 0
register g = 20
register h = 0
register i = 0
register j = 0
```

Here are the diffs for this simple fix to add this new feature. Fixes source file main.c (function "pswitch") and manpage nroff.man. There are only a couple of lines to add, so I did not think I needed a shar file for this:

```
*** old\main.c  Sun Jul 22 12:28:24 1990
--- main.c  Tue Jul 24 03:43:34 1990
***************
*** 687,692 ****
     int     swgood;
     char    mfile[256];
     char   *ptmac;

     swgood = TRUE;
     if (*p == '-')
--- 687,694 ----
     int     swgood;
     char    mfile[256];
     char   *ptmac;
+    int  indx;
+    int  val;
```

```
     swgood = TRUE;
     if (*p == '-')
***************
*** 838,843 ****
                  p++;
                  set (&pg.offset, ctod (p), '1', 0, 0, HUGE);
                  set_ireg (".o", pg.offset, 0);
             }
             break;

--- 840,865 ----
                  p++;
                  set (&pg.offset, ctod (p), '1', 0, 0, HUGE);
                  set_ireg (".o", pg.offset, 0);
+             }
+             break;
+
+        case 'r':               /* set number reg */
+        case 'R':
+             if (!isalpha (*(p+1)))
+             {
+                  fprintf (stderr,
+                       "***%s: invalid number register name (%c)\n",
+                       myname, (int) *(p+1));
+             }
+             else
+             {
+                  /*
+                   *   indx is the user num register and val
+                   *   is the final value.
+                   */
+                  indx = tolower (*(p+1)) - 'a';
+                  val  = atoi (p+2);
+                  set (&dc.nr[indx], val, '1', 0, -INFINITE, INFINITE);
             }
             break;
```

```
*** old\nroff.man  Tue Jul 24 03:58:52 1990
--- nroff.man  Tue Jul 24 03:58:16 1990
***************
*** 33,38 ****
  no immediate output should be generated from this file (see ENVIRONMENT).
  .IP -o<file>
  Set error log file (default is stderr).
  .IP -po<n>
  Shift output right n spaces (like .po).
  .IP -pn<n>
--- 33,40 ----
  no immediate output should be generated from this file (see ENVIRONMENT).
```

```
  .IP -o<file>
  Set error log file (default is stderr).
+ .IP -raN
+ Preset number register 'a' (single character) to N (decimal integer only).
  .IP -po<n>
  Shift output right n spaces (like .po).
  .IP -pn<n>
```

Enjoy...

-bill rosenkranz
rosenkra%c1yankee@convex.com

```
From 8262428414499776074
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,f7760f23ba74cf26,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1990-09-17 13:24:32 PST
Path: gmdzi!unido!mcsun!uunet!snorkelwacker!apple!sun-barr!newstop!grapevine!
panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v03i095: nroff110pat2 -- Patch 2 for nroff v. 1.10
Keywords: shar, diff
Message-ID: <311@grapevine.EBay.Sun.COM>
Date: 17 Sep 90 20:24:32 GMT
Sender: news@grapevine.EBay.Sun.COM
Reply-To: rosenkra%c1yankee@convex.com (Bill Rosenkranz)
Lines: 1118
Approved: koreth@panarthea.ebay.sun.com
Posted: Mon Sep 17 21:24:32 1990

Submitted-by: rosenkra%c1yankee@convex.com (Bill Rosenkranz)
Posting-number: Volume 3, Issue 95
Archive-name: nroff110pat2

This is patch 2 for nroff v1.10 for atari TOS.

It adds the .mc command (margin characters, useful for change bars).
It also adds some additional special character escapes (e.g. \(br, \(pl, etc)
and cleans up a few things. The manpage is changed to reflect the .mc
capability.

The patch is supplied as diff -c files (*.dif). This is a shar file. the
diffs were done on an ST and the file names have "\" in the path, if that
matters to your patch program...  [I didn't uuencode this, on the assumption
that if you can use "patch" you can probably unshar as well. -sg]

NOTE: Apply this patch after patch 1.
```

Enjoy...

-bill rosenkranz
rosenkra%c1yankee@convex.com

```
------ cut here -------- cut here -------- cut here -------- cut here ------
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# If this archive is complete, you will see the following message at the end:
#        "End of shell archive."
#
# Contents:
#   README command.c.dif escape.c.dif low.c.dif main.c.dif
#   nroff.h.dif nroff.man.dif text.c.dif mc.ms mc.out
#
# Wrapped by rosenkra@cray.com on Wed Jul 25 02:13:25 1990
#
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f README -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"README\"
else
echo shar: Extracting \"README\" \(894 characters\)
sed "s/^X//" >README <<'END_OF_README'
X
XThis is patch 2 for nroff v1.10 for atari TOS.
X
XIt adds the .mc command (margin characters, useful for change bars).
XIt also adds some additional special character escapes (e.g. \(br, \(pl, etc)
Xand cleans up a few things. The manpage is changed to reflect the .mc
Xcapability.
X
XA test file is added to test the .mc command. The file is called "mc.ms"
Xand should be executed with:
X
X
X    % nroff -ms mc.ms >mc.out
X
Xor
X
X    % nroff -ms -po5 mc.ms >mc.out
X
Xto test things out. Read this file as well since it shows how to use .mc.
XThe file mc.out is also included in this patch.
X
XThis command seems to preserve the same bugs from the "real" unix nroff.
XTo use change bars, it is best to cause a break before and after the change:
X
X    ...
X    .br \" start changes...
X    .mc \(br 0.2i
```

```
X    ...
X    These are the changes.
X    ...
X    .mc
X    .br \" end changes...
X    ...
X
XEnjoy...
X
X
X-bill rosenkranz
Xrosenkra%c1yankee@convex.com
X
X
END_OF_README
if test 894 -ne `wc -c <README`; then
    echo shar: \"README\" unpacked with wrong size!
fi
# end of overwriting check
fi
if test -f command.c.dif -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"command.c.dif\"
else
echo shar: Extracting \"command.c.dif\" \(2295 characters\)
sed "s/^X//" >command.c.dif <<'END_OF_command.c.dif'
X*** old\command.c  Sun Jul 22 11:34:48 1990
X--- command.c  Tue Jul 24 22:55:10 1990
X***************
X*** 840,846 ****
X          */
X         maceval (p, macexp);
X         break;
X!   case NA:
X         /*
X          *   no adjust
X          *
X--- 840,846 ----
X          */
X         maceval (p, macexp);
X         break;
X!   case MC:
X         /*
X          *   margin character (change bars)
X          *
X***************
X*** 842,847 ****
X         break;
X    case NA:
X         /*
X          *   no adjust
X          *
X          *   .na
X--- 842,866 ----
```

```
X         break;
X    case MC:
X         /*
X+         *   margin character (change bars)
X+         *
X+         *   .mc [c [N]]
X+         *
X+         *   currently forced to "|", 2 spaces to the right, right
X+         *   margin only...
X+         */
X+        val = getval (p, &argtyp);
X+        if (argtyp == '\r' || argtyp == '\n')
X+            mc_ing = FALSE;        /* turn off... */
X+        else
X+        {
X+            mc_ing   = TRUE;   /* turn on... */
X+            mc_space = 2;      /* force these for now... */
X+            mc_char  = '|';
X+        }
X+        break;
X+   case NA:
X+        /*
X          *   no adjust
X          *
X          *   .na
X***************
X*** 1173,1178 ****
X    if (c1 == 'f' && c2 == 't')      return (FT);
X    if (c1 == 'h' && c2 == 'e')      return (HE);
X    if (c1 == 'i' && c2 == 'f')      return (IF);
X    if (c1 == 'i' && c2 == 'g')      return (IG);
X    if (c1 == 'i' && c2 == 'n')      return (IN);
X    if (c1 == 'j' && c2 == 'u')      return (JU);
X--- 1192,1199 ----
X    if (c1 == 'f' && c2 == 't')      return (FT);
X    if (c1 == 'h' && c2 == 'e')      return (HE);
X    if (c1 == 'i' && c2 == 'f')      return (IF);
X+ /*    if (c1 == 'i' && c2 == 'e')      return (IE);*/
X+ /*    if (c1 == 'e' && c2 == 'l')      return (EL);*/
X    if (c1 == 'i' && c2 == 'g')      return (IG);
X    if (c1 == 'i' && c2 == 'n')      return (IN);
X    if (c1 == 'j' && c2 == 'u')      return (JU);
X***************
X*** 1184,1189 ****
X    if (c1 == 'm' && c2 == '2')      return (M2);
X    if (c1 == 'm' && c2 == '3')      return (M3);
X    if (c1 == 'm' && c2 == '4')      return (M4);
X    if (c1 == 'n' && c2 == 'a')      return (NA);
X    if (c1 == 'n' && c2 == 'e')      return (NE);
X    if (c1 == 'n' && c2 == 'f')      return (NF);
X--- 1205,1211 ----
X    if (c1 == 'm' && c2 == '2')      return (M2);
X    if (c1 == 'm' && c2 == '3')      return (M3);
```

```
X    if (c1 == 'm' && c2 == '4')      return (M4);
X+   if (c1 == 'm' && c2 == 'c')      return (MC);
X    if (c1 == 'n' && c2 == 'a')      return (NA);
X    if (c1 == 'n' && c2 == 'e')      return (NE);
X    if (c1 == 'n' && c2 == 'f')      return (NF);
END_OF_command.c.dif
if test 2295 -ne `wc -c <command.c.dif`; then
    echo shar: \"command.c.dif\" unpacked with wrong size!
fi
# end of overwriting check
fi
if test -f escape.c.dif -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"escape.c.dif\"
else
echo shar: Extracting \"escape.c.dif\" \(1131 characters\)
sed "s/^X//" >escape.c.dif <<'END_OF_escape.c.dif'
X*** old\escape.c  Mon Mar  5 22:17:12 1990
X--- escape.c  Tue Jul 24 19:15:26 1990
X***************
X*** 934,939 ****
X    if (c1 == 'i' && c2 == '1') {*c = 0xF4; return (2);}
     /* integral1 */
X    if (c1 == 'i' && c2 == '2') {*c = 0xF5; return (2);}
     /* integral2 */
X
X    /*
X     *   greek
X     */
X--- 934,950 ----
X    if (c1 == 'i' && c2 == '1') {*c = 0xF4; return (2);}
     /* integral1 */
X    if (c1 == 'i' && c2 == '2') {*c = 0xF5; return (2);}
     /* integral2 */
X
X+   if (c1 == 'b' && c2 == 'r') {*c = 0x7C; return (2);}
     /* box v rule */
X+   if (c1 == 'b' && c2 == 'v') {*c = 0x7C; return (2);}
     /* bold vert */
X+   if (c1 == 'p' && c2 == 'l') {*c = 0x2B; return (2);}
     /* math plus */
X+   if (c1 == 'm' && c2 == 'i') {*c = 0x2D; return (2);}
     /* math minus */
X+   if (c1 == 'e' && c2 == 'q') {*c = 0x3D; return (2);}
     /* math equal */
X+   if (c1 == '*' && c2 == '*') {*c = 0x2A; return (2);}
     /* math star */
X+   if (c1 == 's' && c2 == 'l') {*c = 0x2F; return (2);}
     /* slash */
X+   if (c1 == 'u' && c2 == 'l') {*c = 0x5F; return (2);}
     /* underrule */
X+   if (c1 == 's' && c2 == 'c') {*c = 0xDD; return (2);}
     /* section */
X+
```

```
X+
X    /*
X     *   greek
X     */
END_OF_escape.c.dif
if test 1131 -ne `wc -c <escape.c.dif`; then
    echo shar: \"escape.c.dif\" unpacked with wrong size!
fi
# end of overwriting check
fi
if test -f low.c.dif -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"low.c.dif\"
else
echo shar: Extracting \"low.c.dif\" \(815 characters\)
sed "s/^X//" >low.c.dif <<'END_OF_low.c.dif'
X*** old\low.c  Thu Mar 29 00:29:50 1990
X--- low.c  Tue Jul 24 22:59:48 1990
X***************
X*** 56,61 ****
X
X    if (co.outp > 0)
X    {
X        co.outbuf[co.outp]   = '\r';
X        co.outbuf[co.outp+1] = '\n';
X        co.outbuf[co.outp+2] = EOS;
X--- 56,64 ----
X
X    if (co.outp > 0)
X    {
X+       /*
X+        *   handle margin char (change bar) here for all filled lines
X+        */
X        co.outbuf[co.outp]   = '\r';
X        co.outbuf[co.outp+1] = '\n';
X        co.outbuf[co.outp+2] = EOS;
X***************
X*** 59,64 ****
X        co.outbuf[co.outp]   = '\r';
X        co.outbuf[co.outp+1] = '\n';
X        co.outbuf[co.outp+2] = EOS;
X        put (co.outbuf);
X    }
X    co.outp   = 0;
X--- 62,70 ----
X        co.outbuf[co.outp]   = '\r';
X        co.outbuf[co.outp+1] = '\n';
X        co.outbuf[co.outp+2] = EOS;
X+
X+       do_mc (co.outbuf);
X+
X        put (co.outbuf);
X    }
X    co.outp   = 0;
```

```
END_OF_low.c.dif
if test 815 -ne `wc -c <low.c.dif`; then
    echo shar: \"low.c.dif\" unpacked with wrong size!
fi
# end of overwriting check
fi
if test -f main.c.dif -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"main.c.dif\"
else
echo shar: Extracting \"main.c.dif\" \(5418 characters\)
sed "s/^X//" >main.c.dif <<'END_OF_main.c.dif'
X*** old\main.c  Tue Jul 24 03:43:34 1990
X--- main.c  Tue Jul 24 22:53:12 1990
X***************
X*** 62,67 ****
X    hold_screen = FALSE;
X    debugging   = FALSE;
X    stepping    = FALSE;
X    out_stream  = stdout;
X    err_stream  = stderr;
X    dbg_stream  = stderr;
X--- 62,68 ----
X    hold_screen = FALSE;
X    debugging   = FALSE;
X    stepping    = FALSE;
X+   mc_ing      = FALSE;
X    out_stream  = stdout;
X    err_stream  = stderr;
X    dbg_stream  = stderr;
X***************
X*** 251,256 ****
X    time_t        tval;
X    char        *ctim;
X
X    tval       = time (0L);
X    dc.fill    = YES;
X    dc.dofnt   = YES;
X--- 252,262 ----
X    time_t        tval;
X    char        *ctim;
X
X+   /*
X+    *   misc global flags, etc...
X+    */
X+   mc_space   = 2;
X+   mc_char    = '|';
X    tval       = time (0L);
X    ctim       = ctime (&tval);
X
X***************
X*** 252,257 ****
X    char        *ctim;
X
```

```
X    tval       = time (0L);
X    dc.fill    = YES;
X    dc.dofnt   = YES;
X    dc.lsval   = 1;
X--- 258,268 ----
X    mc_space   = 2;
X    mc_char    = '|';
X    tval       = time (0L);
X+   ctim       = ctime (&tval);
X+
X+   /*
X+    *   basic document controls...
X+    */
X    dc.fill    = YES;
X    dc.dofnt   = YES;
X    dc.lsval   = 1;
X***************
X*** 290,296 ****
X     */
X    for (i = 0; i < MAXREGS; i++)
X    {
X!        rg[i].rname[0] = rg[i].rname[1] = rg[i].rname[2] = rg[i].rname[3] = '\0';
X         rg[i].rauto = 1;
X         rg[i].rval  = 0;
X         rg[i].rflag = RF_READ;
X--- 301,310 ----
X     */
X    for (i = 0; i < MAXREGS; i++)
X    {
X!        rg[i].rname[0] = EOS;
X!        rg[i].rname[1] = EOS;
X!        rg[i].rname[2] = EOS;
X!        rg[i].rname[3] = EOS;
X         rg[i].rauto = 1;
X         rg[i].rval  = 0;
X         rg[i].rflag = RF_READ;
X***************
X*** 298,304 ****
X    }
X
X
X-
X    /*
X     *   this should be checked...
X     */
X--- 312,317 ----
X    }
X
X
X    /*
X     *   predefined regs. these are read/write:
X     */
X***************
```

```
X*** 300,311 ****
X
X
X    /*
X-    *   this should be checked...
X-    */
X-   ctim = ctime (&tval);
X-
X-
X-   /*
X     *   predefined regs. these are read/write:
X     */
X    i = 0;
X--- 313,318 ----
X
X
X    /*
X     *   predefined regs. these are read/write:
X     */
X    i = 0;
X***************
X*** 462,468 ****
X
X
X    /*
X!    *   these are read only:
X     */
X    strcpy (rg[i].rname, ".$");      /* num of args at current macro*/
X    rg[i].rauto = 1;
X--- 469,475 ----
X
X
X    /*
X!    *   these are read only (by user):
X     */
X    strcpy (rg[i].rname, ".$");      /* num args at current macro*/
X    rg[i].rauto = 1;
X***************
X*** 464,470 ****
X    /*
X     *   these are read only:
X     */
X!   strcpy (rg[i].rname, ".$");      /* num of args at current macro*/
X    rg[i].rauto = 1;
X    rg[i].rval  = 0;
X    rg[i].rflag = RF_READ;
X--- 471,477 ----
X    /*
X     *   these are read only (by user):
X     */
X!   strcpy (rg[i].rname, ".$");      /* num args at current macro*/
X    rg[i].rauto = 1;
X    rg[i].rval  = 0;
```

```
X    rg[i].rflag = RF_READ;
X***************
X*** 624,629 ****
X    rg[i].rflag = RF_READ;
X    rg[i].rfmt  = '1';
X
X    pg.curpag   = 0;
X    pg.newpag   = 1;
X    pg.lineno   = 0;
X--- 631,639 ----
X    rg[i].rflag = RF_READ;
X    rg[i].rfmt  = '1';
X
X+   /*
X+    *   page controls...
X+    */
X    pg.curpag   = 0;
X    pg.newpag   = 1;
X    pg.lineno   = 0;
X***************
X*** 648,653 ****
X    pg.ehlim[RIGHT] = pg.ohlim[RIGHT] = dc.rmval;
X    pg.eflim[RIGHT] = pg.oflim[RIGHT] = dc.rmval;
X
X    co.outp   = 0;
X    co.outw   = 0;
X    co.outwds = 0;
X--- 658,666 ----
X    pg.ehlim[RIGHT] = pg.ohlim[RIGHT] = dc.rmval;
X    pg.eflim[RIGHT] = pg.oflim[RIGHT] = dc.rmval;
X
X+   /*
X+    *   output buffer controls...
X+    */
X    co.outp   = 0;
X    co.outw   = 0;
X    co.outwds = 0;
X***************
X*** 652,657 ****
X    co.outw   = 0;
X    co.outwds = 0;
X    co.lpr    = FALSE;
X    for (i = 0; i < MAXLINE; ++i)
X         co.outbuf[i] = EOS;
X
X--- 665,671 ----
X    co.outw   = 0;
X    co.outwds = 0;
X    co.lpr    = FALSE;
X+   co.outesc = 0;
X    for (i = 0; i < MAXLINE; ++i)
X         co.outbuf[i] = EOS;
X
```

```
X***************
X*** 655,660 ****
X    for (i = 0; i < MAXLINE; ++i)
X         co.outbuf[i] = EOS;
X
X    for (i = 0; i < MXMDEF; ++i)
X         mac.mnames[i] = NULL_CPTR;
X    for (i = 0; i < MACBUF; ++i)
X--- 669,677 ----
X    for (i = 0; i < MAXLINE; ++i)
X         co.outbuf[i] = EOS;
X
X+   /*
X+    *   macros...
X+    */
X    for (i = 0; i < MXMDEF; ++i)
X         mac.mnames[i] = NULL_CPTR;
X    for (i = 0; i < MACBUF; ++i)
X***************
X*** 658,664 ****
X    for (i = 0; i < MXMDEF; ++i)
X         mac.mnames[i] = NULL_CPTR;
X    for (i = 0; i < MACBUF; ++i)
X!        mac.mb[i] = '\0';
X    for (i = 0; i < MAXPBB; ++i)
X         mac.pbb[i] = '\0';
X    mac.lastp = 0;
X--- 675,681 ----
X    for (i = 0; i < MXMDEF; ++i)
X         mac.mnames[i] = NULL_CPTR;
X    for (i = 0; i < MACBUF; ++i)
X!        mac.mb[i] = EOS;
X    for (i = 0; i < MAXPBB; ++i)
X         mac.pbb[i] = EOS;
X    mac.lastp = 0;
X***************
X*** 660,666 ****
X    for (i = 0; i < MACBUF; ++i)
X         mac.mb[i] = '\0';
X    for (i = 0; i < MAXPBB; ++i)
X!        mac.pbb[i] = '\0';
X    mac.lastp = 0;
X    mac.emb   = &mac.mb[0];
X    mac.ppb   = NULL_CPTR;
X--- 677,683 ----
X    for (i = 0; i < MACBUF; ++i)
X         mac.mb[i] = EOS;
X    for (i = 0; i < MAXPBB; ++i)
X!        mac.pbb[i] = EOS;
X    mac.lastp = 0;
X    mac.emb   = &mac.mb[0];
X    mac.ppb   = NULL_CPTR;
X***************
```

```
X*** 665,670 ****
X    mac.emb   = &mac.mb[0];
X    mac.ppb   = NULL_CPTR;
X
X    for (i = 0; i < Nfiles+1; ++i)
X         sofile[i] = NULL_FPTR;
X  }
X--- 682,690 ----
X    mac.emb   = &mac.mb[0];
X    mac.ppb   = NULL_CPTR;
X
X+   /*
X+    *   file descriptors (for sourced files)
X+    */
X    for (i = 0; i < Nfiles+1; ++i)
X         sofile[i] = NULL_FPTR;
X  }
END_OF_main.c.dif
if test 5418 -ne `wc -c <main.c.dif`; then
    echo shar: \"main.c.dif\" unpacked with wrong size!
fi
# end of overwriting check
fi
if test -f nroff.h.dif -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"nroff.h.dif\"
else
echo shar: Extracting \"nroff.h.dif\" \(1192 characters\)
sed "s/^X//" >nroff.h.dif <<'END_OF_nroff.h.dif'
X*** old\nroff.h  Sun Jul 22 11:40:44 1990
X--- nroff.h  Tue Jul 24 22:38:10 1990
X***************
X*** 425,430 ****
X  char   *skipbl ();
X  char   *getfield ();
X
X  int    comand ();
X  int    comtyp ();
X  int    gettl ();
X--- 425,431 ----
X  char   *skipbl ();
X  char   *getfield ();
X
X+ int    do_mc ();
X  int    comand ();
X  int    comtyp ();
X  int    gettl ();
X***************
X*** 506,511 ****
X  int             hold_screen;
X  int             debugging;
X  int             stepping;     /* paging */
X  char            tmpdir[256];
X  char            termcap[1030];
```

```
X  char            s_standout[20];
X--- 507,515 ----
X  int             hold_screen;
X  int             debugging;
X  int             stepping;     /* paging */
X+ int             mc_ing = 0;       /* turned off */
X+ int             mc_space = 2;
X+ char            mc_char = '|';
X  char            tmpdir[256];
X  char            termcap[1030];
X  char            s_standout[20];
X***************
X*** 549,554 ****
X  extern int      hold_screen;
X  extern int      debugging;
X  extern int      stepping;
X  extern char          tmpdir[];
X  extern char          termcap[];
X  extern char          s_standout[];
X--- 553,561 ----
X  extern int      hold_screen;
X  extern int      debugging;
X  extern int      stepping;
X+ extern int      mc_ing;
X+ extern int      mc_space;
X+ extern char          mc_char;
X  extern char          tmpdir[];
X  extern char          termcap[];
X  extern char          s_standout[];
END_OF_nroff.h.dif
if test 1192 -ne `wc -c <nroff.h.dif`; then
    echo shar: \"nroff.h.dif\" unpacked with wrong size!
fi
# end of overwriting check
fi
if test -f nroff.man.dif -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"nroff.man.dif\"
else
echo shar: Extracting \"nroff.man.dif\" \(1594 characters\)
sed "s/^X//" >nroff.man.dif <<'END_OF_nroff.man.dif'
X*** old\nroff.man  Tue Jul 24 15:33:02 1990
X--- nroff.man  Wed Jul 25 00:21:16 1990
X***************
X*** 328,333 ****
X  This command affects the footer the same way the \.m1
X  command affects the header.
X  .\"
X  .IP .na
X  Noadjust.
X  Ajdustment is turned off; the right margin is ragged.
X--- 328,341 ----
X  This command affects the footer the same way the \.m1
X  command affects the header.
```

```
X  .\"
X+ .IP .mc
X+ Margin character. The first argument is the character to use,
X+ the second is the distance to the right of the right margin
X+ to place the margin character. Useful for change bars.
X+ No arguments turns the capability off. Note that with this
X+ nroff, the margin character is always a vertical bar ("|")
X+ and this distance is always 2 spaces (0.2i).
X+ .\"
X  .IP .na
X  Noadjust.
X  Ajdustment is turned off; the right margin is ragged.
X***************
X*** 643,649 ****
X  documented.
X  Good luck!
X  .SH REQUEST SUMMARY
X! The following table summarizes the nroff requests:
X  .nf
X  .ne 8
X  .cc +
X--- 651,657 ----
X  documented.
X  Good luck!
X  .SH REQUEST SUMMARY
X! The following table summarizes the nroff requests currently available:
X  .nf
X  .ne 8
X  .cc +
X***************
X*** 740,745 ****
X  +ne 4
X  Miscellaneous
X
X  .pm t           -        all      -       print macro names
X  .fl            -        -        B       flush output buffer
X
X--- 748,754 ----
X  +ne 4
X  Miscellaneous
X
X+ .mc c N         -        off      E,m     margin char c, seperation N
X  .pm t           -        all      -       print macro names
X  .fl            -        -        B       flush output buffer
X
END_OF_nroff.man.dif
if test 1594 -ne `wc -c <nroff.man.dif`; then
    echo shar: \"nroff.man.dif\" unpacked with wrong size!
fi
# end of overwriting check
fi
if test -f text.c.dif -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"text.c.dif\"
```

```
else
echo shar: Extracting \"text.c.dif\" \(4090 characters\)
sed "s/^X//" >text.c.dif <<'END_OF_text.c.dif'
X*** old\text.c  Thu Mar 29 00:47:06 1990
X--- text.c  Tue Jul 24 23:43:40 1990
X***************
X*** 104,109 ****
X          *   centered (.ce)
X          */
X         center (p);
X         put (p);
X         --dc.ceval;
X    }
X--- 104,110 ----
X          *   centered (.ce)
X          */
X         center (p);
X+        do_mc (p);
X         put (p);
X         --dc.ceval;
X    }
X***************
X*** 112,117 ****
X         /*
X          *   all blank line
X          */
X         put (p);
X    }
X    else if (dc.fill == NO)
X--- 113,119 ----
X         /*
X          *   all blank line
X          */
X+        do_mc (p);
X         put (p);
X    }
X    else if (dc.fill == NO)
X***************
X*** 119,124 ****
X         /*
X          *   unfilled (.nf)
X          */
X         put (p);
X    }
X    else
X--- 121,127 ----
X         /*
X          *   unfilled (.nf)
X          */
X+        do_mc (p);
X         put (p);
X    }
X    else
```

```
X***************
X*** 602,607 ****
X   */
X
X    register int  i;
X
X    if (dc.prflg == TRUE && n > 0)
X    {
X--- 605,611 ----
X   */
X
X    register int  i;
X+   register int  j;
X
X
X    if (dc.prflg == TRUE && n > 0)
X***************
X*** 603,608 ****
X
X    register int  i;
X
X    if (dc.prflg == TRUE && n > 0)
X    {
X         for (i = 0; i < n; ++i)
X--- 607,613 ----
X    register int  i;
X    register int  j;
X
X+
X    if (dc.prflg == TRUE && n > 0)
X    {
X         for (i = 0; i < n; ++i)
X***************
X*** 607,612 ****
X    {
X         for (i = 0; i < n; ++i)
X         {
X              prchar ('\n', out_stream);
X         }
X         prchar ('\r', out_stream);
X--- 612,630 ----
X    {
X         for (i = 0; i < n; ++i)
X         {
X+             /*
X+              *   handle blank line with changebar
X+              */
X+             if (mc_ing == TRUE)
X+             {
X+                  for (j = 0; j < pg.offset; ++j)
X+                       prchar (' ', out_stream);
X+                  for (j = 0; j < dc.rmval; ++j)
X+                       prchar (' ', out_stream);
```

```
X+                for (j = 0; j < mc_space; j++)
X+                     prchar (' ', out_stream);
X+                prchar (mc_char, out_stream);
X+           }
X            prchar ('\n', out_stream);
X            prchar ('\r', out_stream);
X        }
X***************
X*** 608,613 ****
X        for (i = 0; i < n; ++i)
X        {
X            prchar ('\n', out_stream);
X        }
X        prchar ('\r', out_stream);
X    }
X--- 626,632 ----
X                prchar (mc_char, out_stream);
X            }
X            prchar ('\n', out_stream);
X+           prchar ('\r', out_stream);
X        }
X    }
X  }
X***************
X*** 609,615 ****
X        {
X            prchar ('\n', out_stream);
X        }
X-       prchar ('\r', out_stream);
X    }
X  }
X
X--- 628,633 ----
X            prchar ('\n', out_stream);
X            prchar ('\r', out_stream);
X        }
X    }
X  }
X
X***************
X*** 808,811 ****
X    }
X
X    return (w);
X  }
X--- 826,937 ----
X    }
X
X    return (w);
X+ }
X+
X+
X+
```

```
X+
X+ /*------------------------------*/
X+ /*    do_mc             */
X+ /*------------------------------*/
X+ do_mc (p)
X+ char   *p;
X+ {
X+
X+ /*
X+  *    add margin char (change bar) for .nf and .ce lines.
X+  *
X+  *    filled lines handled in robrk(). blank lines (.sp) handled in skip().
X+  *    note: robrk() calls this routine, too.
X+  */
X+
X+   register char  *ps;
X+   register int  nspaces;
X+   register int  i;
X+   register int  has_cr;
X+   register int  has_lf;
X+   int      len;
X+   int      nesc;
X+
X+
X+   if (mc_ing == FALSE)
X+        return;
X+
X+
X+   len = strlen (p);
X+
X+
X+   /*
X+    *   get to the end...
X+    */
X+   ps = p;
X+   while (*ps)
X+        ps++;
X+
X+
X+   /*
X+    *   check for cr and lf
X+    */
X+   ps--;
X+   has_lf = 0;
X+   has_cr = 0;
X+   while (ps >= p && (*ps == '\r' || *ps == '\n'))
X+   {
X+        if (*ps == '\n')
X+             has_lf++;
X+        else
X+             has_cr++;
X+
X+        len--;
```

```
X+        ps--;
X+    }
X+    if (has_lf < has_cr)
X+        has_lf = has_cr;
X+    else if (has_cr < has_lf)
X+        has_cr = has_lf;
X+
X+
X+    /*
X+     *   remove any trailing blanks here
X+     */
X+    while (ps >= p && *ps == ' ')
X+    {
X+        ps--;
X+        len--;
X+    }
X+    *++ps = EOS;
X+
X+
X+    /*
X+     *   add trailing spaces for short lines. count escapes, subtract
X+     *   from len. use rmval for rigth margin (minus tival which is
X+     *   added later in put).
X+     */
X+    nesc    = countesc (p);
X+    len    -= nesc;
X+    nspaces = dc.rmval - dc.tival - len;
X+    for (i = 0; i < nspaces; i++, ps++)
X+        *ps = ' ';
X+
X+
X+    /*
X+     *   add the bar...
X+     */
X+    for (i = 0; i < mc_space; i++, ps++)
X+        *ps = ' ';
X+    *ps++ = mc_char;
X+
X+
X+    /*
X+     *   replace cr, lf, and EOS
X+     */
X+    while (has_lf--)
X+    {
X+        *ps++ = '\r';
X+        *ps++ = '\n';
X+    }
X+    *ps = EOS;
X+
X+
X+    return;
X  }
END_OF_text.c.dif
```

```
if test 4090 -ne `wc -c <text.c.dif`; then
    echo shar: \"text.c.dif\" unpacked with wrong size!
fi
# end of overwriting check
fi
if test -f mc.ms -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"mc.ms\"
else
echo shar: Extracting \"mc.ms\" \(1780 characters\)
sed "s/^X//" >mc.ms <<'END_OF_mc.ms'
X.\" test .mc capability. execute with:   nroff -ms mc.ms
X.\" tests .mc on centered, filled, unfilled text.
X.ce 1
X.mc \(br 0.2i
XTesting the Margin Character Command
X.mc
X.SH
XOVERVIEW
X.LP
XThis is a test of the \.mc command in nroff.
XIt adds a character in the right margin when on and leaves if off otherwise.
X.br
X.mc | 2
X.LP
XThese lines should have a change bar.
XTurn on margin character mode by adding an \.mc command with at least one
Xargument.
XNote that nroff currently ignores the argument, except to turn on the mode.
XThe margin character is currently always a vertical bar ("|") and is located
X2 spaces to the right of the right margin.
X.mc
X.br
XNow it should be off.
XTurn off mc mode with an \.mc command with no arguments.
X.LP
XYou can add margin characters to no fill areas as well.
Xnroff will automatically add enough spaces to fill the line and make the
Xcharacter line up with the right margin as with fill mode.
X.sp
X.nf
XThis is a no fill area.
XHere is a change bar in a no fill area (note the blank
Xline is changed as well):
X.mc \(br 0.2i
X
XTHIS LINE IS CHANGED!
X.mc
XNO MORE CHANGES!!!
XIf the line is longer than the right margin, the
Xmargin character is appended to the line:
X.mc \(br 0.2i
XTHIS IS A VERY,VERY,VERY,VERY,VERY,VERY,VERY,VERY,VERY,VERY LONG LINE...
X.mc
```

```
XNO MORE CHANGES HERE!!!
X.fi
X.br
X.mc \(br 0.2i
X.SH
XNEW SECTION
X.LP
XHere is an entire section added.
XIt should have change bars for all lines, including the spaces before
Xthe section heading.
X.LP
XThis \.mc capability contains the same bugs as the normal unix \.mc
Xcommand.
XYou should put a \.br command before the \.mc which ends the margin
Xcharacter section to force the character on lines which end before
Xthe entire line is filled.
X.mc
X.br
X.SH
XFINALE
X.LP
XIt section should not have any more change bars.
END_OF_mc.ms
if test 1780 -ne `wc -c <mc.ms`; then
    echo shar: \"mc.ms\" unpacked with wrong size!
fi
# end of overwriting check
fi
if test -f mc.out -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"mc.out\"
else
echo shar: Extracting \"mc.out\" \(2579 characters\)
sed "s/^X//" >mc.out <<'END_OF_mc.out'
X                                                                    |
X                                                                    |
X
X                                                                    |
X                                                                    |
X            Testing the Margin Character Command                    |
X
X
XOVERVIEW
X
XThis is  a  test  of  the  .mc  command in nroff.  It adds a
Xcharacter in the right margin when  on  and  leaves  if  off
Xotherwise.
X                                                                    |
XThese lines  should  have  a  change  bar.    Turn on margin  |
Xcharacter mode by adding an .mc command with  at  least  one  |
Xargument.   Note  that nroff currently ignores the argument,  |
Xexcept to turn  on  the  mode.    The  margin  character  is  |
Xcurrently  always  a  vertical  bar  ("|")  and is located 2  |
Xspaces to the right of the right margin.                       |
```

```
XNow it should be off.  Turn off mc mode with an .mc  command
Xwith no arguments.
X
XYou  can  add  margin  characters  to no fill areas as well.
Xnroff will automatically add enough spaces to fill the  line
Xand make the character line up with the right margin as with
Xfill mode.
X
XThis is a no fill area.
XHere is a change bar in a no fill area (note the blank
Xline is changed as well):
X                                                              |
XTHIS LINE IS CHANGED!                                         |
XNO MORE CHANGES!!!
XIf the line is longer than the right margin, the
Xmargin character is appended to the line:
XTHIS IS A VERY,VERY,VERY,VERY,VERY,VERY,VERY,VERY,VERY,VERY LONG LINE...  |
XNO MORE CHANGES HERE!!!
X                                                              |
X                                                              |
XNEW SECTION                                                   |
X                                                              |
XHere is an entire section added.  It should have change bars  |
Xfor  all  lines,  including  the  spaces  before the section  |
Xheading.                                                      |
X                                                              |
XThis .mc capability contains the same  bugs  as  the  normal  |
Xunix .mc  command.   You should put a .br command before the  |
X.mc which ends the margin character  section  to  force  the  |
Xcharacter  on  lines  which  end  before  the entire line is  |
Xfilled.                                                       |
X
X
XFINALE
X
XIt section should not have any more change bars.
X
X
X
X
X
X
X
X                        25 July 1990
X
X
END_OF_mc.out
if test 2579 -ne `wc -c <mc.out`; then
    echo shar: \"mc.out\" unpacked with wrong size!
fi
# end of overwriting check
fi
echo shar: End of shell archive.
```

```
exit 0


From -4432172444098041149
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,92a6b15132b2d559,start
X-Google-Attributes: gid100493,public
X-Google-Thread: 111a55,92a6b15132b2d559,start
X-Google-Attributes: gid111a55,public
X-Google-ArrivalTime: 1990-09-20 12:30:53 PST
Path: gmdzi!unido!mcsun!uunet!snorkelwacker!usc!cs.utexas.edu!sun-barr!newstop!
grapevine!panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st,comp.sys.atari.st
Subject: v04INF1: Introduction to comp.sources.atari.st
Message-ID: <339@grapevine.EBay.Sun.COM>
Date: 20 Sep 90 19:30:53 GMT
Expires: 18 Jan 91 19:30:53 GMT
Sender: news@grapevine.EBay.Sun.COM
Followup-To: comp.sources.atari.st
Lines: 151
Approved: koreth@panarthea.ebay.sun.com (Steven Grimm)
Xref: gmdzi comp.sources.atari.st:345 comp.sys.atari.st:26996
Posted: Thu Sep 20 20:30:53 1990

Submitted-by: koreth@panarthea.ebay.sun.com (Steven Grimm)
Posting-number: Volume 4, Info 1
Archive-name: intro

This is the first of two introductory articles about comp.sources.atari.st.
This one describes how to submit source to the newsgroup, where the
archive sites are, and how to contact them.  The companion article lists
all previously-published sources.

I am always looking for suggestions on how to improve the usefulness
of the newsgroup, and can be contacted as listed below.

-- Steven Grimm
   koreth@panarthea.ebay.sun.com

--------------------

Subject: Submitting source for publication

Items intended for posting or queries and problem notes should be sent to
atari-sources@panarthea.ebay.sun.com. If you are on a UUCP-only site, you
can send them to sun!ebay!panarthea!atari-sources.  In Europe, send your
submissions to Jan-Hinrich Fessel, the European submoderator of the group,
at unido!atari-sources (or, if you're a masochist, atari-sources@unido.
informatik.uni-dortmund.de.)

If you want verification of arrival, so say in a cover note, or at the
beginning of your submission, if it is small.  I try to verify that a
```

program works, and if I can't get it to work, I may hold up posting it for a couple of days. Please note that, except in rare cases, source without documentation will not be published. The backlog from receipt to posting varies from one to four weeks depending mostly on the set of submissions currently in my queue.

I prefer that you submit sources and binaries separately, as it's a pain to separate them, and such submissions are invariably placed on the back burner. Sources should be submitted in the same format as binaries -- uuencoded arc, zoo, or lzh files. Please use Pierre Dumas' "uue" program if possible, as it can correct some common transmission errors. Split long submissions up into reasonable (less than 50K) chunks. "uue" will do this for you, if you use "uue -700 file" to encode.

Please inform me of any special system requirements, and what development environment was used, so that I can properly test your submission and label it when I post it.

Subject: The structure of comp.sources.atari.st articles

Each posting in comp.sources.atari.st is called an "issue"; there are roughly 100 issues to a volume. The division is arbitrary and may vary. There are two types of articles in comp.sources.atari.st: sources and "information postings." They can be distinguished by the subject line:

  Subject: v04INF2: Index for Volume 1 and other info

This first word in the title identifies this as the third info posting of volume one. Similarly, the subject line shown below:

  Subject: v04i081: godsrc -- Universe simulator

identifies this as the 81st source article in Volume 4. Large sources are broken up into smaller pieces, and have subject lines that look like this:

  Subject: v04i088: decbustr -- VMS clone in ST-LOGO, Part03/08

Certain information about the system configuration required to use the program is given on the keywords line.

  Keywords: MWC, Megamax, 1meg, high

This means that the program compiles with Mark Williams or Megamax C, requires at least one meg of RAM and only runs in high resolution. Following is a list of keywords; new ones may be added as needed. They are mostly self-explanatory.

  Alcyon - Alcyon C (included with Atari developer's kit)
  MWC   - Mark Williams C

```
  Megamax - Megamax C
  Laser       - Laser C
  Lattice - Lattice C
  C       - Any C compiler
  OSI         - OSI Personal Pascal
  BASIC       - ST BASIC (this is not a joke!)
  uuencode    - program is uuencoded (uudecode required to unpack)
  arc         - program is archived (arc required to unpack)
  high        - high resolution
  medium  - medium resolution
  low         - low resolution
  1meg        - needs 1 meg of RAM
```

The first few lines of an article are auxiliary headers that look like this:

```
  Submitted-by: david@bdt.UUCP (David Beckmeyer)
  Posting-number: Volume 4, Issue 80
  Archive-name: micro-rtx
```

The "Submitted by" is the author of the program. If you have comments about the sources published in comp.sources.atari.st this is the person to contact. When possible, this address is in domain form, otherwise it is a UUCP bang path relative to some major (backbone) site.

The second line repeats the volume/issue information for the aide of notes sites and automatic archiving programs.

The Archive-name is the "official" name of this source in the archive. Large postings will have names that look like this:

```
Archive-name: desktop/part01
```

Please try to use this name when requesting that sources be mailed to you. Also, note that the "part number" given in the title, and the archive name given in the auxiliary header need not be identical.

Since most archive sites run UNIX, articles are given UNIX-style filenames rather than ST-style filenames. I do try to limit filenames to eight characters whenever possible, though.

Subject: Reporting and tracking bugs and patches to postings

Most of the time, bugs and updates will be posted to comp.sys.atari.st. Sometimes, "official" updates from the author will be posted submitted to comp.sources.atari.st so that they are archived with the sources.

Since there is no easy way to apply source code patches on the ST, changes to a program often result in a whole new version being posted.

To report bugs, contact the person listed in the Submitted-to header. Often there is a contact address in a README file, too. I do not maintain

the sources I moderate, so don't send your bug reports to me.

Subject: Listing of archive sites in no particular order

1. Site panarthea is running an archive server, which automatically answers mail requests. Send mail to archive-server@panarthea.ebay.sun.com with the word "help" in the message body for more information. Both the sources and binaries groups are archived on panarthea.
2. Unido is the European archive site. It's running an archive server similar to panarthea's; send "help" to unido!archive-server for more information. Unido archives both the sources and binaries groups.

If you know of additional sites which reliably archive comp.sources.atari.st (or comp.binaries.atari.st, for that matter), please tell me so I can add them to the list.

From 3876303878669277254
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,34d532f22e40eefc,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,34d532f22e40eefc,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1990-09-20 12:32:54 PST
Path: gmdzi!unido!mcsun!uunet!snorkelwacker!apple!sun-barr!newstop!grapevine!panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st,comp.sys.atari.st
Subject: v04INF2: Index of previously-published sources
Message-ID: <340@grapevine.EBay.Sun.COM>
Date: 20 Sep 90 19:32:54 GMT
Expires: 19 Nov 90 19:32:53 GMT
Sender: news@grapevine.EBay.Sun.COM
Followup-To: comp.sources.atari.st
Lines: 312
Approved: koreth@panarthea.ebay.sun.com
Xref: gmdzi comp.sources.atari.st:346 comp.sys.atari.st:26997
Posted: Thu Sep 20 20:32:54 1990

Submitted-by: koreth@panarthea.ebay.sun.com (Steven Grimm)
Posting-number: Volume 4, Info 2
Archive-name: Index

Note: When requesting files from the archive server, replace the last "/" in a file's path with a space, i.e. volume1/make part01. You may request several files from the same directory (volume1/make part01 part02 part03).

-- Volume 1 --

volume1/intro Introduction to comp.sources.atari.st
volume1/basic-uud Uudecode in BASIC

```
volume1/bootcopy        Automatic file copier for RAMdisk
volume1/joyread              Joystick-reading routines
volume1/cc/part01       C compiler driver (Alcyon)
volume1/cc/part02
volume1/cc/part03
volume1/cc/part04
volume1/mkmfor               Something for Absoft Fortran
volume1/make/part01          Program maintainer
volume1/make/part02
volume1/make/part03
volume1/make/part04
volume1/make/part05
volume1/make/part06
volume1/make/part07
volume1/blank           Fast disk eraser
volume1/mmdummy              Megamax dummy routines
volume1/fdum            Megamax floating-point dummy routines
volume1/idle            Screen saver
volume1/autotime        Time setter for auto folder
volume1/time            Time setter
volume1/autodisk        Copy files to RAMdisk
volume1/dvist32/part01       DVI previewer
volume1/dvist32/part02
volume1/dvist32/part03
volume1/dvist32/part04
volume1/dvist32/part05
volume1/dvist32/part06
volume1/dvist32/part07
volume1/dvist32/part08
volume1/dvist32/part09
volume1/cache           Disk cache
volume1/vdit            VDI interface with GDOS font support
volume1/flip/part01          Double-sided disk copier
volume1/flip/part02
volume1/rzsz/part01          UNIX end for zmdm, below
volume1/rzsz/part02
volume1/rzsz/part03
volume1/rzsz/part04
volume1/rzsz/part05
volume1/zmdm/part01          Zmodem file transfer program
volume1/zmdm/part02
volume1/zmdm/part03
volume1/zmdm/part04
volume1/zmdm/part05
volume1/zmdm/part06
volume1/zmdm/part07
volume1/zmdm/part08
volume1/ops5/part01          OPS5 production system in LISP
volume1/ops5/part02
volume1/ops5/part03
volume1/assist2/part01       Multifunction desk accessory
volume1/assist2/part02
volume1/errorchk        Verbose display of error codes
```

```
volume1/joinprnt       Echo console output to printer
volume1/color          Color-selection dialog box
volume1/rsctoc              Convert resource to C source
volume1/fastdump       Text-only screen dump
volume1/readme12       Online help system
volume1/newstart       Fancy printing
volume1/varlint
volume1/flip27/part01       Double-sided disk copier
volume1/flip27/part02
volume1/poplib              Shared libraries, with examples
volume1/uupc/part01         UUCP clone (pre-pre-release)
volume1/uupc/part02
volume1/uupc/part03
volume1/uupc/part04
volume1/gnufp          GNU floating point routines
volume1/scroll              Fast screen scroller
volume1/break          Stop programs with a keypress
volume1/newbreak       New version of above
volume1/timecode       Determine 68000 code speed
volume1/achlib/part01       Memory management and other routines for MadMac
volume1/achlib/part02
volume1/sttar          Unix-style "tar" utility
volume1/d110edit       D110 patch editor (MIDI)
volume1/raymovi2/part01     Raytrace movie generator
volume1/raymovi2/part02
volume1/gstlink             Faster linker for GST object files
volume1/mx2v230/part01      Multitasking kernel with utilities
volume1/mx2v230/part02
volume1/mx2v230/part03
volume1/mx2v230/part04
volume1/mx2v230/part05
volume1/mx2v230/part06
volume1/mx2v230/part07
volume1/mx2v230/part08
volume1/limiter             Limit uppermost hard disk device letter
volume1/d110edit       Patch editor for D110 synthesizer
volume1/mt32edit       Patch editor for MT32 synthesizer
volume1/starchrt/part01     Diffs to Unix starchart program for ST
volume1/starchrt/part02
volume1/passm/part01        Logic circuit assembler
volume1/passm/part02
volume1/leolife             Game of Life

-- Volume 2 --

volume2/sc_src/part01       Source to Sozobon C compiler
volume2/sc_src/part02
volume2/sc_src/part03
volume2/sc_src/part04
volume2/sc_src/part05
volume2/sc_src/part06
volume2/sc_src/part07
volume2/sc_src/part08
```

```
volume2/sc_src/part09
volume2/dlibs12/part01      Source for DLibs C libraries
volume2/dlibs12/part02
volume2/dlibs12/part03
volume2/shar            Pack and unpack shell archives
volume2/keyfix              Fix Gulam for non-American keyboards
volume2/sed/part01      GNU sed (stream editor)
volume2/sed/part02
volume2/sort/part01         UNIX-style sort utility
volume2/sort/part02
volume2/uw/part01       Unix Windows windowing terminal software
volume2/uw/part02
volume2/uw/part03
volume2/uw/part04
volume2/uw/part05
volume2/uuduue              Dumas uuencode/decode programs
volume2/mwstring        dpANS string.h for MWC
volume2/mailtruk        GFA BASIC uucp software
volume2/format83        Make Intel ASCII Hex files
volume2/mailtruk.11         Version 1.1 of GFA-BASIC uucp software
volume2/1stjet              1st-Word configuration file for DeskJet
volume2/1stjethv        1st-Word config file to use helvetica font on DeskJet
volume2/cpboot              Copy executable boot sectors between floppy formats
volume2/du              List disk usage in directory trees
volume2/fplib10             Floating point routines for Sozobon C
volume2/prtset              Set printer parameters
volume2/hexify              Hexidecimal encode/decode for mailing binaries
volume2/nutshell        Use a shell from within a GEM prog
volume2/texchang        Change files for BibTeX, Tangle, and Weave
volume2/deskset             Re-read DESKTOP.INF and set up system accordingly
volume2/which           Find programs using PATH env. variable
volume2/atime89             Set GEMDOS time from keyboard clock
volume2/fplib20/part01      New floating-point library for Sozobon C
volume2/fplib20/part02
volume2/decky           Connect DEC keyboard to ST
volume2/djreset             Reset DeskJet Plus
volume2/modedit             Hardwire an editor into German Modula-2
volume2/overscan        Software driver for increased screen size
volume2/gnutar/part01       GNU's version of UNIX "tar"
volume2/gnutar/part02
volume2/gnutar/part03
volume2/gnutar/part04
volume2/gnutar/part05
volume2/gnutar/part06
volume2/gnutar/part07
volume2/gnutar/part08
volume2/coloremu        Color emulator for mono screens
volume2/gccgem/part01       GEM libraries for GCC
volume2/gccgem/part02
volume2/gccgem/part03
volume2/gccgem/part04
volume2/gccgem/part05
volume2/gccgem/part06
```

```
volume2/postmort       Analyze post-bomb information
volume2/crunch              Executable file compressor
volume2/environ             Set environment variables from AUTO folder
volume2/viruskil       Virus killer
volume2/norad          Earth satellite tracking program
volume2/diff           File difference utility
volume2/emacs.diff/part01   Diffs to GNU-EMACS 18.52 for Atari
volume2/emacs.diff/part02
volume2/protect             Virus killer/protector
volume2/ff31           File finder version 3.1
volume2/gcclib2/part01      Patches to GCC library
volume2/gcclib2/part02
volume2/gcclib2/part03
volume2/gcclib2/part04
volume2/gcclib2/part05
volume2/gcclib2/part06
volume2/gcclib2/part07
volume2/gcclib2-patch1      Patch for GCC library
volume2/mandel              Fast Mandelbrot set generator
volume2/gfadiff             File difference generator
volume2/nolock              Disable Caps-Lock key
volume2/builds09       Build Motorola S-records from .ttp files
volume2/colorca             Cellular automata in color
volume2/eternal3       Reset-proof ramdisk
volume2/degadesk       Load DEGAS .P?3 files onto desktop
volume2/truchet             Display random Truchet tilings on mono screens
volume2/yaff           Yet Another File Finder
volume2/strings             Find text strings in files
volume2/curses/part01       Screen updating package
volume2/curses/part02
volume2/sozobon1.2/part01   Update to Sozobon C compiler
volume2/sozobon1.2/part02
volume2/sozobon1.2/part03
volume2/sozobon1.2/part04
volume2/sozobon1.2/part05
volume2/sozobon1.2/part06
volume2/sozobon1.2/part07
volume2/sozobon1.2/part08
volume2/sozobon1.2/part09

-- Volume 3 --

volume3/fract          Monochrome Mandelbrot generator
volume3/uurename       Rename binary postings to work with uud
volume3/uustuff             More uud-ing tools
volume3/uustuff2       Addendum to previous posting
volume3/xxed/part01         Hex/ASCII file editor
volume3/xxed/part02
volume3/xxed/part03
volume3/xxed/patch01        Messed-up files from XXED
volume3/nroff/part01        Partial clone of UNIX "nroff" formatter
volume3/nroff/part02
volume3/nroff/part03
```

```
volume3/kbext410        Keyboard remapper
volume3/cal             UNIXoid calendar program
volume3/dhrystone       Dhrystone benchmark program
volume3/quix            Game: fill up the screen
volume3/3demo/part01        3-D graphics demo
volume3/3demo/part02
volume3/xxedpat             Messed-up files from XXED
volume3/tosvers             Tell TOS version number
volume3/lharc/part01        LHARC archiver
volume3/lharc/part02
volume3/sound           Sound generation docs, plus sample sounds
volume3/necp6           Screendump programs for NEC P6 printer
volume3/file/part01         Determine file types
volume3/file/part02
volume3/barcodes        Metafont file for printing barcodes
volume3/mx2net20        Share disk drives across MIDI/RS232
volume3/splitit             Split up and reassemble big files
volume3/bm              Fast fixed-string searcher
volume3/abzmon              Machine-language monitor
volume3/metaconv        Convert EZDraw/UniTerm metafiles to GEM format
volume3/netsrc              The REAL sources to mx2net20, MIDI/RS232 network
volume3/szadb/part01        Debugger for Sozobon C
volume3/szadb/part02
volume3/dstools/part01      Data structure toolkit
volume3/dstools/part02
volume3/sam             Disk analyzer
volume3/sre             Run programs when gulam is started up
volume3/lharc113/part01     LHArc archiver, version 1.13
volume3/lharc113/part02
volume3/lharc113/part03
volume3/jove/part01         Emacs variant, v. 4.9
volume3/jove/part02
volume3/jove/part03
volume3/jove/part04
volume3/jove/part05
volume3/jove/part06
volume3/jove/part07
volume3/jove/part08
volume3/jove/part09
volume3/jove/part10
volume3/jove/part11
volume3/jove/part12
volume3/xxed11/part01       XXED editor, version 1.1
volume3/xxed11/part02
volume3/xbra            XBRA trap-catching system
volume3/wd              Uudecoding program (takes multiple file args)
volume3/gfa-uucode      Uuencode/decode program in GFA BASIC
volume3/sozoinit        Startup module for Sozobon C, for accs and prgs
volume3/fgrep           GNU fgrep
volume3/belltest        Test out NewBell bell sounds
volume3/chaos           Draw random Sierpinski Triangles
volume3/bifurc              Draw bifurcation diagrams
volume3/vbi             Vertical blank interrupt handler (w/example)
```

```
volume3/sunclock       Watch the sun go around a map of the Earth
volume3/norad/part01        Satellite tracker, version 2
volume3/norad/part02
volume3/st_conc             Concentration game
volume3/mkrsc/part01        Resource construction program
volume3/mkrsc/part02
volume3/mkrsc/part03
volume3/mines          Game of Mines for mono systems
volume3/showspec       Show Spectrum .SPU pictures
volume3/switcher       Pick random NeoDesk/DegaDesk pics, NewBell sounds
volume3/maze           Walk through a maze in 3-D
volume3/nroff1.10/part01    Nroff clone, version 1.10
volume3/nroff1.10/part02
volume3/nroff1.10/part03
volume3/pi1tomon       Display .PI1 pictures on a mono monitor
volume3/gulwrap             Make Gulam work under Gemini
volume3/dublvis             Format DS disks that work in SS drives
volume3/grep15/part01       GNU grep and fgrep, version 1.5
volume3/grep15/part02
volume3/grep15/part03
volume3/xmore          Configurable "more" for various file types
volume3/mutate              Disk sector editor
volume3/xxxx           Insert numbers into files
volume3/sozo1.2pat     Patches for Sozobon C 1.2
volume3/unx2st              Convert ST <-> UNIX end-of-lines
volume3/mtprog1             Multitasking in Programming tutorial, posting 1
volume3/nroff110pat         Patch for Nroff 1.10
volume3/lharc1.02/part01    Lharc source for ST and UNIX
volume3/lharc1.02/part02
volume3/lharc1.02/part03
volume3/cat2asc             Convert STDCAT43 catalogs to pretty ASCII form
volume3/nroff110pat2        Patch 2 for nroff v. 1.10
volume3/nrchbar             Generate change bars in n/troff documents
volume3/stturing       Two-dimensional Turing machine
volume3/hdwboot             Wait for hard disk to come online


From -7238217256179961554
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,6042ad4a73432e13,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 156df7,6042ad4a73432e13,start
X-Google-Attributes: gid156df7,public
X-Google-Thread: 100493,6042ad4a73432e13,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1990-12-14 13:09:21 PST
Path: gmdzi!unido!mcsun!uunet!cs.utexas.edu!sun-barr!newstop!grapevine!panarthea!
koreth
From: koreth@panarthea.EBay.Sun.COM (Steven Grimm)
Newsgroups: comp.sys.atari.st,comp.binaries.atari.st,comp.sources.atari.st
Subject: Administrative: New feature in comp.sources and comp.binaries.atari.st
Message-ID: <677@grapevine.EBay.Sun.COM>
Date: 14 Dec 90 00:02:11 GMT
```

Sender: news@grapevine.EBay.Sun.COM
Followup-To: comp.sys.atari.st
Lines: 8
Approved: koreth@panarthea.ebay.sun.com (Steven Grimm)
Xref: gmdzi comp.sys.atari.st:28911 comp.binaries.atari.st:1314
comp.sources.atari.st:370
Posted: Fri Dec 14 01:02:11 1990

Comp.sources.atari.st and comp.binaries.atari.st postings of more than one
part now have References: lines, so that thread-following newsreaders such
as trn and nn can easily select all the parts of a particular program.

"   !" - Marcel Marceau
Steven Grimm   Moderator, comp.{sources,binaries}.atari.st
koreth@ebay.sun.com   ...!sun!ebay!koreth

From -6543099155383237975
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,bc2c6eb7d3bc7eed,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1991-01-24 10:16:19 PST
Path: gmdzi!unido!mcsun!uunet!pmafire!uudell!bigtex!texsun!newstop!grapevine!
panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v04i056: mgif11 -- GIF viewer for mono systems, v 1.1
Keywords: arc, uue, C
Message-ID: <sources.4.56@panarthea.ebay.sun.com>
Date: 8 Jan 91 21:37:46 GMT
Sender: news@grapevine.EBay.Sun.COM
Reply-To: rosenkra@c1yankee.convex.com (William Rosencranz)
Lines: 1033
Approved: koreth@panarthea.ebay.sun.com

Submitted-by: rosenkra@c1yankee.convex.com (William Rosencranz)
Posting-number: Volume 4, Issue 56
Archive-name: mgif11

mgif 1.1

This is a new version of a GIF viewer for mono STs. It fixes a serious
bug and adds some new features. yes, it is slow, but u get src so u can
fix it. New for v1.1:

     - Change -i to -r (reverse black/white)

     - Change -q to -i (inqure)

     - Started to add -q (quantize), not complete

- Fix bug with zero'ed accumulation arrays (in dither routine) which was being done with every call to rasterize causing poor (i.e. incorrect) F-S dithering

- Add histogram for each rgb value (eliminate the "intensity order" histogram, too)

- Fix scan_CM (bug in goofy Alcyon/GEMLIB read)

- Make default -s of 40

- Histogram now impacted by -N (NTSC)

- Add -f option for filtering color "planes"

This is a GIF viewer for mono STs. It uses Floyd-Steinberg dithering to represent color images on bi-level devices (see Burger and Gillies, "Interactive Computer Graphics" for the reference I used).

There are 2 posts: binary and source.

It handles all image sizes, but clips to max of 640x400. It should handle up to 8-bit images (which is max GIF format, I believe).

It can rotate, mirror, invert, and expand images. It can also change the relative intensity of the image (in the dithering phase). There is an inquire option to just print info about the image (size, bits, etc). It will also plot a histogram, but does not do any image manipulation (such as histogram equalization or color quantization) beyond the transforms listed above. Use PBMplus for that.

Mgif can also dump the resulting image to a standard GEM .pi3 file.

enjoy...

-bill rosenkranz
rosenkra@convex.com

Binary manifest:

Name	Length	Stowage	SF	Size now	Date	Time	CRC
MANIFEST.BIN	549	Crunched	44%	309	7 Jan 91	2:36p	703F
MGIF.1	4659	Crunched	40%	2799	7 Jan 91	1:11p	B06E
MGIF.TTP	33404	Crunched	37%	21277	7 Jan 91	2:19p	3BFC
README	3936	Crunched	39%	2411	7 Jan 91	2:30p	C771
4	42548		38%	26796			

Source manifest:

```
Name          Length    Stowage    SF   Size now  Date       Time    CRC
============  ========  ========  ====  ========  =========  ======  ====
LINEA.ARC         2471     --       0%      2471  17 Dec 90   9:36a  484E
MAKEFILE          1934  Crunched   44%      1092   7 Jan 91   2:39p  8DE6
MANIFEST.SRC       774  Crunched   45%       432   7 Jan 91   2:40p  3E53
MGIF.C           63739  Crunched   49%     32594   7 Jan 91   2:10p  4257
MGIF.MAN          5127  Crunched   42%      2990   7 Jan 91   1:10p  48DF
README            3936  Crunched   39%      2411   7 Jan 91   2:30p  C771
TODO               377  Crunched   24%       288  17 Dec 90   7:05a  B482
        ====  ========            ====  ========
Total      7     78358             47%     42278


---------------------------------- cut here ------------------------------
begin 644 mgif11s.arc
M&@),24Y%02Y!4D,```#_IPD``)$5C4Q.2*<)```:"$Q)3D5!,"Y/`````/]#
M````D1463,S87`````Q@-`"`-$#"P`4<!BIP8@4``"3G`(`#!8#)MP,`G'AQ
M4B>)FS1T!A(`-G``Q2]LTK@I$P8&`%$##6`D.+/`S`$8#P`:"$Q)3D5!,"Y3
M`````/]G`0``D14)3#G)V0$```PJ&BA0D8!-&C=EPL``T0*$G#)TZLAQ`X+-
M&S=G0,!Y<Y!.&3D@Z+P!,8>.G#ICZ`@,.##!G#QNWL"9DV:.CI4K$\`Q":('
M"!0E3Z:L>+#,ES`@5*0@BE#ASQ0[<"H0F,1-&CH]2`2!`6F`0($)7)RQ*(9-
M@B\&F\+XJB"L1SPJIRI`6U3AS;9IUR0(8X/%"!@)NJH@22?,F#4@S,@)TZ8,
MV#9O[)1IX\*LPA9A9+!H@2+,C10)$@R>$T:RPS)GYH"X4P;$&+)B/OXL@V=,
MF9T@R,!(P38!F3$N[B2H>I7MX(T=98LD+)0.B)AW0!S,O3",&S(@%+K('H.A
MC^QR%N<!\<9,XHO.T20D\W$.B^PRO(,7OY*\^;0)$]=QDS+-Q3F]0289969U
M]MD*+%R666B#/532&P^=EAI8^[&A%U^!#3"8&6G@09AAB"G&F&-R)2`''7,$
M1L!@NDFG&G)N>"1';RZ4<1U;&@A,24Y%03$N3P````#_=0```)$5%DQ*H;H`
M```,8#0`@#3`P\`%8@8J<&(%```0;@`@$(%&(HF*#&2X`@"!Q$8&H`(X\>*D
M3A(G5$X.``!L(`$$4*A,44FP)20"#.#0D3.P@,V7>`8>^,D@C]"?#L:\8?.&
M9TV7$+ZP2>.F3)@8`$0--'!@95>"7P>$'=NSZP$:"$Q)3D5!,2Y3`````/^-
M`0``D14)3+,-+0(```PJ&BA0D8!-&C=EPL0`T0($G#IT'*;!4X:-P(`#$\S)
MX^8-G#EIYNBX>#&!G3=IR(`PB%`A"!1SZ,BI,R8BRS)?PH!0`4<F"Q!X?N;Y
M.>8-FS=R4NP@J>#BRH,)0<2<61/$&S-FYI2A(U*@P"1.J(#M@4,@%"I3QL:0
MX74@B#!RSHA\>4=.Q)AAQJP!849.F#9E',I\:R-%VYYRR`K$TV.MP#R-:0@L
M>C1Q#!MM!29P<>:H&#8)OMQ4V'8SG3)XZ+05#57A2`4%#ZY)$,8&BQ$P2K9Y
M8Z=,&Q>@P\!@T0)%F!N&F\+>W1MX`L3&"[,0GB"!BJ<MI<JD&1&Q9N9EG(,5
MZ\0XC!338WSG'1[TV;3EA:,/PW8Y>Q=W$E!&&AV]\1@I5'?=0Q'M)P<(!R'H
M!AT)TA'&9V6LUQQH>/3'@G$R!&B=0Q"!4!12*B78$TAN@.`@A*61,09^"9`0
M!`PPQ)``)`-<1X9?=TA$D47*$03>;Z`9A]P*T^4&6QUNL#%;;:79-4=FL+E0
MAAMDM`4:"$Q)3D5!,RY/`````/_T````D1463,7PO`$```Q@-`"`-(#(0`4!
M,!QT8@4``!!N`""(P0=````8*%H$H$'C10X>`7B($5$A")(`+``0@9("`!(A
M(C(`8"(F``@`5(`:X,2+DSI#A"2A`F/@`&`#"6`(.C2&4:20"&A@2D7&TZ0<
MJ,ZX&M4#DRE4F"1QXA`J`1!,G#0),F5)V:0BKC1Y0J0(5P(DL#B%9,"LB2Q[
M^R9%@<4J7[,JLA@6')4%'#IR!A8PBP#/E[V3DS+(<UFR60>6#6>."H&S:+,2
MVKPA4R:I60IMPLQ9XSJIA3>//2?%\(5-&C=EPFP5-=#``:/'"28OL+RY9.>0
MF#^?#NG`\0,:"$Q)3D5!,RY3`````/^$`@``D14)3&@OV0,```PJ&BA0D8!-
M&C=EPLP`T0($&3EA[H`PB%!@P($)YN1Q\P;.G#1S=%BTF,#.FS1D)AY,N!#%
M'#IRZHRAHQ+AES`@5,"!R0($GA@]\P#U*2-H41!MWI`IT[--F#EK>G:DDV+'
M2(QUYI294[,,B"`@Z+QQ"%$B3HI>S<AYT\8G4*%AQ^(IFD>&BZL6N^)\&7,F
MB#=FS&BE$U*@P"%"DE"!T4,&C<.)J<1H;`.R8AF-<5BF,J/'#!@"F4RAPB2)
M$\\R0CMI$F3*$L^/%5QI\H1($<^5%6"9/$.S@BR3:8#6C9E&ZM_%8QL>""*,
MG#,A08!`<4<.S9=AQJP!H39,&Z\[Y32WD6)Y^!Z^\7R9'.-XGO4]8L16CSE&
```

```
M[O?U?2==VGBX4ZB-'3=58\HID(`+9[#QAAAL)/`%6@HM=R`=9>!!QW(/KJ20
M2`92M$8"8=C`P@C#"91`4G:4X4*#X:$08@HLA`%#`@FHT-57(!R4!DTNPL"0
M#R#P)1,==<A11GD*F(BBBG<D,`)0B"D&0X\IT&@C'&R$@1`(/KK(!AM(GOA&
MBBXT^20+44I&I94@8*FE5S%,-\8;"HKWAAMLY!'FDF4Z"65D,JQ98YM9;BG#
M=`J"8"17=^:YYYA,^HEF9#,(>F6A7K64Z*)_X:FGDI#V.96+Y+$@&FFFK6GC
M0Q%-]!1-<*2!1QEL_&5&5S^`2F:3_ZU!*HQ,K-;:$JK>V.MT<]")4@]&0$+`
MH[N*N=2O+,Q66Q'%5K>C5_MYA8*1;HY11@\P0*MB@^K%0.UNEH(PYQMRI`18
MFR>Y09BNYR9`W[J!RHBDB62,T2<)0<``PPP)0#+`JF5U)6$=>'X8HH36S2&A
M"V6X0<9R&@A-2P!%03,N4P`````#_70````)$5!TPNP(4````,(T"T";.FC)DT
M;,J`H/,&A)@Z",F`8)/&39DP(-Z(45-F#)TY#12$#,.&C8X$"2A:#`/#14.5
M%V.XG%CQX@R7(4/"9.GRY,Z6(!7LE/G&9\TP,H/NO%DTY=&;00$:"%)%041-
M12Y,00```/\\`0``D16-3(>9HP$```Q&WL@!00?-FSEE0-Q)4_!-'3H@V*1Q
M4R8,"#D.Z4PL,X<%030<$X:1DW#.FS8)"X:!F$0A1CHBW9!1B";-&#0-%+2L
M@Q#$1!!MSJ0QXP*$D#IIV,PLF/"-&#5EQM"9`P)%FS!K$K8P`W1-"A!A9((E
MLQ1D&X)O0#P1HF2*3S<Y@PX%BK6,F:1EBCX9&(;LQS)GZ:2U*%&,G)%YP(J]
M&?9,0B9)A+C]*9?KU:QWV>3-F=/)&Y@Z_H)PD38S1Q`@28)5'83-F#QOW(`P
M(_`J1!1W:MX$,<=@':5@V9A4*'"-0H8X%4QYH\=I;(\-ZYQ!`Z(EFC!V$KKY
M3)`CS+*`4Q0-./!S:A!C3L+!*Z=CQ#19032Y,H1%3MAUCK-A`X)BF:5ID20&
M4L`Q!8)34$E%E1D8G652'7*,L9D".0$:`!H(34%+149)3$4`````_T0$```G
M%N%TYHV.!P``#"-`M`FSIHR9-&S*@##S1H[`,VG,-%`P8F)`,F'HE-$!`A*!
M)F_<@%`21B2(&R!@Q-`Q@X8.&31`Q,B1(X9%$';*R)F3)B3'&"YL4IPX<4J1
M(502).@ALV@5(4:1*NTQD4H0*4>*)%WZ,*(+.G3@3&P2Q`F4(%FG=C7C@@2*
MJ%12N)WR%&Z*B4."4#F;EFL;B&S'9!Q;=HJ4(6K_>AWH9N+5(4:2,"F2&+"+
M,'+&%$5R-?)DOY;GH)$SL4J5(IXI@_9:ITZ9B4B*!"%21,H4M44/WUX=&,2<
M,G34M('C8L[$)T*4[&:JF.T;$"-^!Q_NXLUQ*K%MJY53)@R9-@H'%CR8$(1;
MLDX,#[FK@$B2*512WV;J]@GVVE-2F'^KFSU>(TR@M=Q$3!`!H(!JM3`'""VX
M(9$"3"0!58)LL)&&&]U-)$023K@GA5ICZ,`%&V\(Q@878EQ(6(=)?,A5B%PP
M%F-)<QT5UT125&'654U,A:..3U2Q%54*$%4D12"0D09W8]#1DQMSN`#)`!9-
MI"233H8TAPX3*05$&6.@\1P=><"AD`CB*11&A2*`0,=S8M2!$!W[68655NQY
M"::8;I9Y9IH@7#@''6NRT>:;(,0YYPN"$EIAG5=E=:,">H8YII\@H$F00AH-
M>NB890P*Z9V35LHGF69F"N@8"97T*0C<M?%&3B"\(88:8-)A'*4)?&EIGZEJ
M6A`(K-HJADZOQCJK0K;BVN2")9$QJJ1Y]KKGI<$"BMD8KP(ZQQMU9*9F9FBD
M0:M;CZ56K:^G8BJL0J)AUNVFOH$KKF]H8+800@K-Q9D4ZG9I[:^H_DEO:V7,
M.RS";HSQ!AEE2.N6G=3N9QIJDA7AWY$!#72AFYB=`9R45`ZE0*$Z*#5QI'@:
MN3*I*:1<7W+Y">S6$.OM5^"!1^3'X',O5SRS<OJY%2%4&]_$&!QAB`P""G7\
MEJB<;-#1PL<N,.:"8'3*44="*9!<I0)+-[U1`N>5Q9?&+J.`WMHQHXU"7GNA
MQ3:O#K,QAAQTWJP7W"#X0`=U7`M<AAN$BN$FX1GM][;=U<K1QN+$%7ZDWW5G
M%7?:Z1V6-,=N/N>U2&G0(;;)G=*1LMP4M\QKT'CNE^..4C11-`JS!SFID0$A
MVFBA@9H^Y=B_5YAR\6Q\D2+I4#J:/&-&(J_\A3*CT'JI"8P!Q[2QN[4ABU+(
M93W+NRL@/6/5/ZZYP-H[KC;D[H,OOOIW\UXK'6CH%.7P)K/:G1M<XI7D]H,<
MHH%`)SO9#]W@9J1BB>%8<DB9_THBL`'"+B[Q8^"1MA5`E:'@,?H)PW[2E;';
MN0<^\EG7M9(4DO`0Y$)G&.%A`G8DA'50;A=SPA">0)O;74^%OR)#"P7R0C?$
MT"T7H^%$XA5!F[WE7R9\3WPRYC,?^*LS)108NYXC1`P1<0TPW,\4_J5$T$',
M3&Z`6,/2$*K3581LEGG#ZIJS-2,!&@A-04Y)1D535"Y34D,`L`$``"<6%G53
M/@8#```,3L*T*0,"D@(F9=R<H8.F(($I=-[<"7.&(*2'1@H.F))&#T$W$D&`
M(!*&CL4#5-(,=#A$RI`&"GI`8B!2)H*:D&Z"D$D`ITZ9"7`:P$D`)I,D3HH$
M<1'$9<$$,FC<B%&P0(L6!0_`*%'00-2I(F/<&%EF#(@<,#3FT#'#1AB1-'#0
M*`*S29`E18PD85*DH((8.6;0$#E$3ATW8]"4(:.1!@VND`S$@)%#AL:Q2L*X
M.4L5T@`9;'/`$8F#2!$;=8,X26*DR!0J+J8XA73`1HX<A`TC5LS8,XT:D`_,
MP#'6,V;-G#6"I@%C-(@;0:#7/<+:Q9""VS,N#$#-XC"AQ,O;IP#<H$9,FKD
M&&P<1.;-.3I_UC'9.8WT-Z97M^O$;XT8,HSUG6[B]3;`?9!Y=5M:[;V7G&<Q
MT-<<7*49`9,42A'11%\S===6;N'QIE%W"485@WS'P2<?:#-,^-T-4\%$Q1-$
M/%'0`]L)"-YNXWD657`RX(!#6&,14=99#`YP@PXPU/`6"$+$)0-,!2``DT\%
```

```
M,7#E3CG)^`8=8;#157$%W(!#D#,4U(!4YMTGPPPSP`0:"$U'248N0P!%`
M`/]2?P``)Q98<5="^_@```PCZK@A4\8,B"E)G!QA4N2+D20,$T`:\$(%"#)O
M0+QQPR8/B!@@S*1A4\;%1!`J7C10$'!@01!5IC24245)$R@2*5JL,Z<,"#8;
MSZBITP8."#H9SY2A`T),F#%KCF9L$R:-&Y0J60HD:'#*D"!,OA`I8B4AE9P5
M0[Z1`X)(&3MIQI1)XH9.F:MA0HZY^L:@F#1TYF!=*;%`VC)L>H(8LW$.G3!U
M!RM867$ER@1GTAAL$1)H'C(M')>Q*J:,G#,7`:,Q#0)%FXUO4H#(F]E@&JI*
M6[NHG<*R"M^8-5\LPY@@6S::[9()*>=-&Q!0A#2!PX8G"#A/UX11ZB(D&=6L
MTPANX_'-G:L8K9YQ`1RH&Z%%+[XI(]C-&Z9WUD9%D6?ICQ3L*8`2<&B(AY0<
M>>C@&W`YP/!"##(\:,-'+L``0G,]N;&&'&$DP)E5@*41!AL7(E9&&(JA`,<;
MHBV'U&(NS.%"&(_)D4:,=/0FX&\[)I!##`^^<`.%(&%XUX8=<C8&&I#EUD(:
M4H'0@AS`04*`DDR^YU,+<43Y9)57@N!8&'(H%V489"S'96MQU`$9'6GH448*
M+(!@'U.,%46276!R)A(>3=6!VAVJ@2!G<R>4L=Q38Q!5!QLTIK'1;')PF,<<
M8-Y1X)(@W(%B4Z.])Y\;/A%*!QH@O&6:1V.,2")28'(HFHUR]CE;FB`4Z-@;
M9W#XG!EKI?H4JJ>)`8(=(];A$PJ(W685C66`>:I/(EAEEQMS`%:>',:)D*N!
MO/I:)U*QV4K5&CY5T9,35'@%PAE`.<6&K7^*V:H;7PS11&MB"`J"5>^6T09R
MQLIQ(ADZ7F8E9^?ZQ%48CS(5FD8&T0"#K;HBU6L8S]EWQ[]%/66F&!ZUX$1K
M['J5L`H2A8FFF@:]`0><&_F6U4ICPCG&8DRRI<(<8XR1[7(]@"`"$"B,(%L;
MM1$)PH]!#FFDAAQR`8/5($`2@0@[K+0DF5DCH`)Y;G#L4]$B,*V9U6Q?[;;5
M7'O=<]@JV&%:MI.BK;9!,;@`$M1`2LWBD56_;2$D$\0]V>(C6#5&=02!P(-C
MW[WA`AH^K-2X&X_7$3D/+/XUT.69LZ095S#)]`5--D&A^;^<0^[3Y$NI413I
MFM_UG1DK]:Y`6M,J=H=I/ED,20&S#02"###7`8*5D(W!EE4]-!(%%$DT<D4`"
M.#A?`!/;IW7;=LO*:I<<LO5KAAFFU4FR78*EI+GTI()0/192!''%]L9_'[Y%
M'/I8<11U$1KE17WLDX/[\@`_Z-'/)T@(PA20@!`M\`\&.;#!_\04)Y_T)5<H
M0M5CQ$"2^&4E>@69'@B>4(5V)<&""8@!#)BWP6S)B6(:J0,=X*##HX2!A#Z1
M'TL>"`(D)&$*+H0A\S28@+38T(,&R5BX..9#(#HPA?6K'A0JN+TEUK"#.&0,
M4-A"%:.,D"17%$D6K0>%)&"!"0DI0@)LT+P-4@50;B#*==*`!\0(!F#?F0-U
MPN"1YMRA3D(L3%I(14`:_>1$C@&!#6@@`[6PA4R62DT;3#@_+%+/>E_`PO8F
M"8,-YK$-I6'+!^'`1S]JY"I`,]A=ZH2&M<1)B"A4XR>Q\(4L7+"43;3(*5.)
M0U;V,3&OM)<LW5`GNY4)ET1DB!'.D@!@IN4[TV++=PPV!IIA*XTJE$(2CH`$
M:L;`=[D,9Q&(``4F!,$)<CSG$#T)@B-(H0A%<$([WRE'&712ER`0`A.J4(1]
MPC,!-/BG"L'"!(/*$07W9*<[X0D"/M3SGOET:$4#.M""3K0(.E)H_?))A(<P
M!`5F2$$"BO#!$!2ME,#+"!TX!!40%.$)1DBF57C(%)&@48B^HT,>X*"HEP@D
M6V<@U7*^QI8Z,/4+=.B:`H1*5-0=-0U))":"U0%`':T%5JE0MJD&NFM7EN`<U
MCPK*5X,Z5+&*::9.I0-:+!++,MQE.+%,P\R")<0]$,9:"9C#%\B`!Q9L<'RY
M>:)DMB=8,N1!J@D`K&#%N)8YU"F8(!C!8MXP1DXJ@+%?^$M@#(M9T<[A!<9$
MS&(#^X4QR&$'&Z0L6PPV!\[J4%)7221367N&,;2!M&F!UQODM=DQ@J",G=G.
M:@4KAJ22`;:ES4ZOWK"5XO+U9@KH`V-=:U=\196M57V)8^00U[F"C'QX=:U>
M#R09OWX6L&GX0F$WN*)L>;-$K\I('>!`U.,4A"F)C.P7\L`"P@B8L,"E*Q@#
M'%_'%OBS`I:M93%[2NO*X8]706P0L[*]^)H6NFDQ+6I;22+=]DS`O?WM!GGB
M$^$25[;'#8-1&/P%:T%*+K!-"PQZT),VW05.(]((6T*P8VN9YL8$!&IV.XR;
M,JQU<;ZC#(\NX^(1S4%!.\)NX\R`NIC,I`@UN8F4;0>'T-;!#-M30E&*X`8[
M[$"(:7T/5-.\YC9O(08UZ()41Z`[S?C.J3V;<P*D@*+S;>%^V#N"GC&KQB";
M+SQ-KM,0.-L$&<..L))9":#)).A)$X0,1TB"$0YMO?Q=0<]I"2`(F*"%*VSV
MTQ<J-&O"8(;S<=4-)'P#5)1\UNVY!2YRH<NU<FP1@@#;)T;&EK8D$^<S"%H(
M@'G"^GK"E)<2.Z"`6>&TEX)#4N$!3V^(G!`!FP`L#`$$+TWP"G7(4Q!@(7G+
MR<)FUT(&P7RP497Z\1Z/N=HLG-O:*R$W$D9S!C300=T:YN`-`WR%-)#A5%(E
M-Q10)!AK8U9=!"19#G?8P^;<MG[",;(<D+P<5NYLW'79WA-T.&F!5!L$,+BV
M6_($!]K65I7L[N$)QE13QKC\!)(AM["/_)0R7-L,D$(-L-@B<I*?-S<HEVL"
M/%V&*730L&FANL++4">#H8DYSJFGJ'-U,-8$F"Z`H;K5Y81UBTRA1N".W!/K
MQ.+ED*$.-A+5$$@"M@!37=U62)9/#$8'O"NU*84\6&O#O6$#5QH/?\<L<MJ0
M;615QR=+7PS>#1:9`6Y]M7L_D1PBGY90Y_1QHG]UXR%\4R.0WB*F3]5`6M"7
M)[EAZ52YK^?]CO?7MX4XSJDY?6Y^+#*)"(B>53D;R/!W`].E^1`V0AHN3`<C
MR++M()`^]9EC5]7722EN.$')61-[01)GM=:WJ^]G'GP%VJD,W^:^3^8`%*8`
```

```
MC$ES$.$/?\KA!$C?#4/0,]C'?D7!%LAB(_NW>ML#;710:7.P!MCW3D008P]H
M207T&)\78%)P,`ZX!M>F=1)(@5&1>?BF>AF8%1)G,'^R!1$T0174!2R0%OBG
M?U94=XAW%*LQ'#3G,_TW!6>V@BU(02^D9P&7<@FP<G1`=5O`0DF$:A9!71S'
M%)A$2%S5$\NA<=.B@^T7=$;(@',`!:8!!22&?0E'(S.5!OW20'['66M1:7"P
M=D97A%+'@!UX=,%"$#LX?+C%A5)G!:8A!H.#;C!W;:<B'O\B&%90!%(@;\X$
MB#UA$@/`A]M#%VTR?6<SB)AE%95H,$)6(B[!@W*X/5'@)G4!1A:7%FWR)F#T
M(C$P(?F')O0AB0F`!.`BB#&'65*T,<^AB5V%(++(4@9QBA;1>CKE!NV&=,J%
M`D<`9D,@&W#V5%)W!/$R:6/DAJ2V14.X!3/@A/44+T$&8\CUC($F==1HC5KT
M@MK(C>!H:<-3A<M29202+$#1*FR0`D)45W<Q9U/@6FY`%V9P;?"H3-V57C:R
M5Z"H`$+79$Y`%-<F'&T0,6GP)$T6$B.Q+'?BC@@C&;TVBP;"@D?4A->6B[ZB
MD4&Q/7)`BX[AD4CT@E*UD6>`DG2@DB"Y$ALI!C`IDRP9BN56!?QE&DSP7[:(
M?6-2)NJQ;ZJ%`@\3,8<(<RP``\[8?UG`D_WUD[5FBQ%GA!B7,O^&B6FAE:F2
MBO>5>=5!>;F%@D8H?6QP/B!F$9G7$R31300D89FV.)71(_,89&8@$-V$6X*!
M`@!#;ZSQ03)&5&0"&7*1`E@FBSR!7BB0`E<I=6000#4VD8WYF":I>(_&%I6I
MDUY'!HL7.9N)D$;X,E^06B02FG;P!@YGDK)&*\OBF#I9&P=R!L:"FJI)!AV&
M+8_!!FS06L@5FN0&-)"Q.A-HFZL96&N@5U]0!$+0&K`IFE(GG/B2!,7YG*EY
MG-*9+_L"G$:8G=43:N?&G5+'9>"AF<]);I$9!G?P!8;DG&LY&S4")4C4&LA1
M/T'PE#J9GNN9,>Z9GW+`G^*Y/1CQ!?7YFN]I0#9R$%1`GPN%G]"Y/?A&?)6Y
M00@JGPN*`DK1!AAA6<="!S4@`PY*;FU@!ZV%=Q+JF!0:GPK:&G;@H2`JBX\C
M!U^`CU<QH9A5H2N*H0*SH<WDH@ZZD0,*)VU@H"EJ?#F:H3PJ)I=R@"%JA'=@
M(W91FFDP`_WYH`FP),2Q!E^`+AYAHVF!H_.)`D*0!$\P!4TJ=9X"&%M:!EV*
MHC>JHF$ZIF5ZIMNSC%00GL\9972I`BT`"1X@1!61`%1A%3D!J'SJIT(TJ#5*
M)KU5)XQJ!SH"6(PZ!E+%5%DS`"KPJ%M`A`K@7E(&'%20@\#"F^91E"-1'>,%
M+8*1A>!G&GA9D3]!2%"(0RX0>SY%'XDY92RS`GUZ`2L`'!;%JXBS`A4%K!LU
M$;%G=4E%(WCG$Q-A47R0`*:'`S?0(1.!`@:SB8JR,@D0K'TZ`<3*!\8JK.!:
MK#T"K0?!77?E%GEED!L5K4/R/O11)[HC&*:"*C!02MA:!MJ*,./ZK>'ZK\-J
MKI>!KM(X7$%&C<'BAB#PK@DP`RH@`UY@6H@'/]N#`C+C34&6A@);KM'Z/#SA
M)O78L0%[KAM5!-]V%WAS%4(PCU'QK!65`"&0$Q@[,[@U(MSJK0,KKCW2'3Z[
M/5E#`#X+`@%2L"!`KB5KM!8%`DDPD>RJ7NX*K0E@6->:K;VXK21+L"RCLQYK
MK$O+!+J6L&S(%@SKL%92LQH[LB:+M%K;K4<+L&V+KE9":+/2%@84M%)[ME9K
MB?ZZMG#+LY<QM"X`M%8BN)<*K)QA);%'!:9!EC02+!/1`A]+`/O:K]PJK+ZZ
M(%GF.Y&5<K6T?#/*CW;(%I][A06Q%H.W%(87)03Q@$@Q8QS6:X@A8U9X;;,+
M!U8H)L`W$,D'6`;C3#WQGK][-SY!0MGQ`IH"&`KHN_<!+<+;O';Q=#ZQ`B_0
M`@TR'*B12(!%>96R%N_)O<W!=!.IO2FG&4_QGLEF7QXAG&AD!B(3+##0`C)D
M(>0K5X!(!TC1!N\YE%+(%/>;OXL%6'<!*:=A=(QVJ@)Y5_4KJ&&`+JPT`]<&
MA>V6%P],D?SW5RE7&U_`9>]YJP2Y7M=%&)8Z$2J`NY#5:VF0!I`%6&I`>.^I
M*8Y$;616(H4G!]^TP")!?4)ZB3%PPB5I</?AP^^1``7".9`UPICZP&4SI->V
MQ%!$M!7LP4I&&'69`"AQJ=0V'.X;,3(R$82Q6&AI:Y!P`$7#4`X%6;\6%W-1
M%W<EB/++!)!5NJ%KPT%;-#U,&,WQ&-$[Q@#W6<-[8<Y*QH-(&.`+N8(<<X3Q
MOV''QX/\60/,J,YJ`'W,P`X\I5DCR8T<6>Z[,Y"0`$5C,<(+21N1@,5W>8NE
MQ&9SR46#`I:J`K+A!%7`!'"LIU3,(U9L$1.!'8!,*6?0Q0/PQ8DTJ2W0`HYZ
M&G:P`BL`6<F+1B@PJ2!@`B;0&BJ0J<:,;D5S`BT`='Q@4=/\J-8,`B>P`B>0
M`CJ2`.[580:!`MU<S3UPS>+\S-$<`N/%.?&ASH]:)R*P`@TB`G4R`^1,&!.1
MQ]`BB`T"60G@1\B6SNM\!G;PS=@\S@9VSA,Q!Z;"*>ILS\:\`C'PSY\U$1(]
M`*VB&">0`R>@`QMDO19"$*@AT/>52%;"TI>(TI!E)6+@=1[(N1,1TCYQ`F)0
MTC4$=_#9%/<!P"Y-`(K\''8\TT9MTU(%T""-(CM=!CZ-68]<P%'B4R1"HXME
M)56=&TD-T$M](C>].#FAT^`<!R4-UL)QT=3,T"OPHBX%<^5\/*.HBC=4-(\+
M)1CMUB"JU`@-T`50UZ5XUQ]A`TI=TV+=U!W]U"(M!U.=:JJRR\8+%<A;(-%;
MU'^L&%_=T6'=P(I=UE`-SFWPV!91R.*+7D5MVH)XQYR-V)Z-TXR]TTY`VB#@
ME6/YE6[2TAQF)5G9+EO)VBWCVF/MU&:]<[2=OLO&OICWO@?9,N:[,WB-%'K=
MUL?<UV`MW)^=TZ%]`F9`VSYE:T4=QJP1W:HIS8_ZUL\9W$P-V\6-![3-OT9)
M(H!2U%@@E3X)E.0]W>=MW:V]WF2MW2*=!^\-U*8)`AY1U%'9DW)`E2^7U^:=
MT?RMWHG-WMM-!K1M;&H,.]>BOEM-`&D<;&Q\%:M\5K+AX'M=W>E-T_Y-W-N=
```

```
M!L?M!MKZ2AW1X91XM<A\V"N^V,7]!K0MP3U$P9:<V@U<!@]\XV`MS,3,R\><
MS&J=SL[,`\LSUP0@T05@!C5G+>E,.::A0`!M`")0`E=V7#S!%.8W!IKA$1Y\
MKU+R!ES@!OS<Y61C-BE>`(N9&RBPT4K=!V"-RD,JB-2-XQ/^W[$-SG9`VXT8
MB#[.%"B0B(L8`@[JW.DLSS-%ST9QXOC<`L#+SR!0`QS=,E1^Y761SE\^!VVN
MZ<"+6W,.?]F&`DZ9YV#MAW+@B&5@Y/T=Z"PNTFA`VR+),2^P&FP`NTW>&I)N
MP[Y5Z=1]Z;X.!YK.Z5).Y76^+*T.UGK.V3#Y!K3>,C5NB=>NXK:^V$F9EEC6
M,L_>&M%>ZZ^]V-.^/>F>`$A>S&[-Y)\U[7H:3,!\J0TC)LW:*3[!)'83U'P7
M21N!>14)B<#,86O]Y!]1S@DP[G>>IU#V655\Q1.Q='(!.S$N'(?^B%[\60&<
MSK`NZPJ?[09#Z_,>\;@\`%8Q7G'%EZTA1EUG'@"R\;=L\)$N\CX!S2`0`A\_
M..5\SO0.81*/\FX0(B.RBD7<RRW/=VQ!H^.RNGUA$!&Z%G4"(G1@8$$OI.JA
M\*L5HW,\D*$)H28:+*R>W@D0I+<![0Z_/2-:HA<F]C)4)S+DF`T+ZG0@ZD%0
M:Y"&7H:HG\PT]S:7/![!I2)`]FL?]9KY]A^!YWZ/Y:TA`IGG;?9'F<'R(JJ>
M(]V1!=2U&)`Q^`9=^&%_^#`]WV]^*'>^")'7C/CB<91E(X$&1ZQPYVB&IQO
M8)[?]J`/]_Z\`Z1?]XWO^B#`I2XP^Q!6^Q**^*TH]]>1=[R/`B(`!7_O^UP:
M)8Q1BGE4$A,A_`:6IG2PIFT*6Q.1%MIO@7G!I<(W![Z\6/+^\#]_RY<*%#*3
M3`:CJ$5YJ[Y<\(2QS,N"Y#VO.:?S$@BA$`SA$!!1!`:VUB%@\W,]`,[<QWPV
M$'Y&UNO/_A-/6ZA"HP0/]`Q6\S!^M?"<\SO?$W,-TFP@HUH=FIR[_NP_$3)S
M5[=J]0&&NWY^WO".SJV!H9JQP42C$42%:ZV!NW7BE+(!Y4X9T4YMY<J?Y71@
M'.XWZCJP68^R'!<I,W<%YJ7^8(41YT-:)[B;XB"]$7!2_0:=_@96Q>P_$?<^
M!_F>;89(&\+AP2@%JZL!BTRWJKH"B5K/8)'NG4$`GJVAP5R&S]-:K8.O\!]M
MY<J?Y71@'.XWZCH``F!N)_<Q&^\B'+=:Z@]6&'$^I'6"NRD.TD"1(AK,92D.
MTAL!)]5OT.EO8%7,_@%],`F,'O0!M0="\$#/8)&>G5-0G!K,975B`OMHP_[8
MZ>;LU%:N_%E.!\;!Y9P]ZCI@P3X!YA>8%[CG2%L>+`"CU:T+M0=2Z@_6,G$^
MI'6"NRD.TD"1(AK,92D.TAL!)]5OT.N^UC:O\#Y/]Z*.EF6@`UEC`6!>ZG6"
MNV1/]Z)>ZF1/]Z*^C\L$.\"B`Z5.]G0OZJ5.]G0OZEE3`/NX3+'_<*B"`HV%
M![+!R"5`!J5>)WY9%[*QCS;LCS$R6'A`]G0OZEE3`/NX3&2'50;7&HV5![(!
M"9A<`F10ZG7BEW4A&_MHP_X8(X.5!V1/]Z*>-05PL,151C_0&H*58K+!R"5`
M!J5>)WY9%[*QCS;LCS'R!2E&]G0OZEE3``H[6_1A6_>%`I.%/I=:-"5`!J5>
M)WY9%[*QCS;LCS'26NACT'0OZEE3`%XX8OR&`LP%&',@&XQ<`F10ZG7BEW4A
M&_MHP_X8(Z$%&'-`]G0OZEE3``OY''*)`I,UMG,@&\=3-"5`!J5>)WY9%[*Q
MCS;LCS&R>)U%]G0OZEE3`$(@71ZG/#"&`LSE7+)1-"5`!J5>)WY9%[*QCS;L
MCS$26LY%]G0OZJ5.]NEO8%7,_@%],._BC22RCD8A'.W([QY$*@0/]`R6SOMH
MP_X8(U^08@K_T6MM\U).]Z)^L"\VMC$&!SK0YJ6>]A-]+]K9&AK,90_6,NEJ
MP_X8(XO760_6,BT_-Z[<C<.EL'+@AF2?_@96Q>P_$1CQ+XI>B()!<401BQLQ
MXR"1</UEP5PW&R6"(J,,1%9_\A1'%#:[$3'_RT!_=@JI1Y-<]F\`56??&GC.
MN64OF0F'H9JQP60P]9O<=9'=$_3*$9!<)Z;]\GH\)XJ]95VF.JPC9A"VUCGL
M&#NL\!^=PXX!^Y,\T;53SVH&!VP&J08][7PV$'Y&U@<-*;A+0$5C]GV.`A%^
MI4"1(AK,90YO8%7,_A/QI,H[&U!LR1Y<&DLW?^IZ%4H1&,<U?`2T7P0/]`Q6
M\S!^M?`<8Y5,I3BO(E/JQ#F_RJW\RK',!)UNSDX-6$^Z%-D]`$_*;47SI,HK
MI52J(E/JQ"F.\NG\I$LQUP,@T01`]W;O!H'/II!8ZBE.`-K/_56:$^D^$7^]
MV!)-`,0O]J&_/#20X@1`]Z(.`E>0QS[11E3JP6"NZ7P^)S,]`.N>_@96Q>P_
M$<(Q%)'$B]/G$6J^Z[^5_%NUZ\\Q+5<Q_54/]`R6SC8/ST4$+@K_T70OZFU>
M[;JH`VU>ZBD^``0OZ@U+%P0!*%9B!(]"(I!@``I+MI8&"0:P@<L!"09P!,MT
MR2VK+)=:[;I8ZL]9$2$P$3E1Q-"]/#4P(2_0&@HK!VX(AXYIJ"$P$3E1Q-"]
M/#4P(3[@`]@6&&`H!V)X3&^VVP-0Q-#]$3,]`)F''D#\<J'_+U!B;?\")5!N
M1"OY0KJ?PD=;-$4\!G,]`!)-`$#,%"M0-#>9PIS:,G0OZ@U;`C5`!B4@0\MW
MJ24P0X`R$24P0X`R$24P0WA0`C*T?"4@0\M7`C*T?*7^8(7AEW61`BG\8(5Q
MDRD,@X#MEW4A&^4H8UN0PEVP!3``@X#MEW4A&^4H8UN0PEUP9S`(V'Y9%[)1
MCC*V!2G<!5L@`S`(V"?9D2D,@X#]DAV9PC`(V#;9D2G<!2D^`.F>R\HOZFW>
ML$BQFR6P?/%-ZFY0ZK2$%#DRTR"]$7!2_0:=_@96Q>P_$<(Q%)$$&1;?BP2V
M6=2O+%$"^19L_;\,]`R6SC8_UR"]$7!2_8IM8%7,_A-1&DOG,.(Q2'E0E%@O
M*F1@0'42_IDW_FQ*\$`?8-K/_57*N>O/_A-!$N"W'*S,62]_!S'_RT`?8,0O
M]DRY/#2`_'0OZA9<!CH``B70`C.0`X)Q/)J.NX1/HH;?&C1@`W4B`S2`_'0O
```

```
MZCNL`R!0`C2P?+H[!G5B-/0**;B[K9U/HH;?&C8@`W4B`S2`_'0OZM#/IJT!
M^3$_`,*_/>N#JJB"`I0#A6F?`-K/_>ZY0>&?>>//IN5__E,,857,_A.QZ\]!
M'?=A]0Q6\S!^M0K_T5ROU5\_$<3O]G`O]W0OZI=:[;IH@87X1TVF`Z6>]A.A
MGU\`H&D_$<3O]C50)S+0ZLG/^,R??8_"!L*?$\0O]DS9L+F?_(S/_`V;K\*?
M$\0O]DSY$3D@]W0OZ@UK!$8@_#E!_&(_`S90)S+0ZLG/^,P/`AM(!L*?$\0O
M]C2``W4B`ZV>_(S/_/6T3,*?$\0O]C8@`W4B`ZV>_(S/_`%U>=B_V)F'`D#\
M<J'_+U!B;?\")5!N1"OY0KJ?PL@LY4#,%"M0-#>9PISJU,0O]A]1)S(P^G0O
MZ@U+!<U+(J8I&"6P?)H.Q#DRTP-`_&)O`S)0)S)``W)/]Z(._6P*B5ES`,*?
M$^N#JJB"`I0#A6D_$=K/_>ZY0>&?>>//IN5__@&6_@96Q>P_$4K!%)V)YL$"
M^=)+\$#/8#4/XU<+S[P\!@K_T5ROU5^?R\HOZB"P@=0&)Z)R1O0!B:6>]A/!
M@-)F!EEL)9,\$>:6-0LPR1/A;UFS`),\$1-G_EGCR9D\$4C8<I%Q/),\$6H'
M1L<SR1.!=DG(>'!HE4ZM=9T\R1/Q>%H'"9B,R(L=>F2B=58RR1/1>EH'"9B,
MR(L]!+W'>*J<R1/Q!,NG=9"`R8B\V,_'>%ESR#,]`-KG&.GGK`0PR1.1?F6@
M=<<SR1/Q?P$(-I"`R8B\V'2((B^+R8B\V!N()AV8-04PR1-!ARCRLIB,R(L-
MC%FC`'WLU)GGEU!B;?\")5!N1"OY0KJ?PL@\UP,@T01PDRG<!58)UB?9D2G<
M!58)UB_9D2G<!59YW3>9PEU@E4Z=[A,O]BELE?\")5".:-FC^RF,S%).MX:6
MPEU@E4Z=>7X))=;V+U`"Y?=C:KJ?PL@LY50':J*V!2G<!5;IU)GGEU!B;?\"
M)5`>A!6D^RF,S',]`!)-`,Z?0GBP!2G<!58)UCZX/GRT!2G<!5;IU.D^\6*?
MPE;Y+U`"Y4Q80;J?PL@LY4BHA"G<!5;)WJL!%=Q?I>I.&.E?R_6>TWQW%?L%
M;ZF"!X!A_^^5SG(<2U>!\R%@\_L__"1J^*UA`S)0)S)``\A/]Z(.^+_/IJU1
M^3$_`,*_/>N#JJB"`I0#A6F?`-K/_54*]K;?&GA.&.E_T'BPZN6>_KY3$7WZ
MIUD1J,]>J%E1$7WZIUG!\/`W0)&:<O`W0%+EJ:;#9?Z?$`O1$"95!%*6`%:N
M_%E.!\;!Y9\UZB"P!4_"!RV`!EVP!;1GP670!5L0&F[0!5O0`F3@!EVP!2U0
M!EVP!5/2!5O`)5VP!2W0!EVP!2;3!5N`TEVP!?H,`UVP!2V`!U?1`GG@!EU@
MP650Z@^F]GG@Q,^92%:N_%E.!\;A?J,.`EN0L7S9!19<!J5>)W$^I,_)9XD1
M+;_#,E:N_%E.!\;!Y9\UZB"P!4_"!RV`!EVP!;1GP670!5L0&F[0!5O0`F3@
M!EVP!2U0!EVP!5/2!5O`)5VP!2W0!EVP!2;3!5N`TEVP!?H,`UVP!2V`!U?1
M`GG@!EU@P3X!B:7^8&J?!T[\G(EDY<J?Y71@'.XWZB"P!1G+EUU@P3X!B:5>
M)W$^I,_)9P/A9Y]EY<J?Y71@'.XG`J6>WE:N_%E.!\;A?B+P)%ES`+QHB;,!
MB#V4<!O1$:6>WE:N_%E.!\;A?B+0`J@RQKQHB=>!JM^R*[K8&BV`*K=!'6D0
MBT^2`J6>WE:N_%E.!\;A?B)`>Q;L$QLWP5!LR6E>*$YLP650ZNEMY<J?Y71@
M'.XG`J%Q%9!@`&5^YO92]**RR0<"]U$ROZUA,4OY[3E2ZL\9J%:N_%E.!\;A
M?B+0`F1P%9!@`&5^YL-Q;!JNLLN&E`4!,6FYE!![YSS``Z:5`J7NF(EDY<J?
MY71@'.XG`BU`!E<!"090YF<^',>FX2J[;$A9$!"3EDM)1TY9ZNEMY<J?Y71@
M'.XG`BT0R/"''<HS`S(``W@@`_DJO5>-P#1:ZNEMY<J?Y71@'.XG`E.2-0>0
MV<5[8U$!&<N1O'91ZNEMY<J?Y71@'.XG`FLRQJDXV#[!BA,B8:6>WE:N_%E.
M!\;A?B+0`F9P%9!@`-_-&BCP!%*`0S'0`P9#!BP@`SW0*]W%`C30`R2D+"E0
MZNEMY<J?Y71@'.XG`BWP'&.LVAI6ZNEMY<J?Y71@'.XG`B:3-0?`8K7MV[@=
M*9,BED0Q/4!S%W)1ZNEMY<J?Y71@'.XG`BA]R3`MO4^3TF6`&H^C:VM`*#U1
MZNEMY<J?Y71@'.Z7S]<+"08`T]+[-"E=!JCA<VPL!X^C:VM`*#U1ZNEMY<J?
MY71@'.XG`BV`!U<!"0;`OT59X!]D2YE1-B2B8:6>WE:N_%E.!\;A?B+0`GEP
M%9!@`/Q;E`7^0;:4&65#(AI6ZNEMY<J?Y71@'.[G^+!Z/'FQ&\)QJZ6>WE:N
M_%E.!\;A?B)0ZNEMY<J?Y71@'.XG`DE@$&5^YD7Y&IQX*I#Q2@)/$N.R&AYQ
M(ISRBISH*F)2T:A";7#R'C)2ZNEMY<J?Y71@'.XG`E0P%4.>/-);0'*`+N['
M8C]A'JQA><HB&)D7&F[@`J6>WE:N_%E.!\;A?B)`MT69A9:G+#C4`F1P%?B'
M@_.7R@7!/MTT&X(!%,.3=Z@1&F[@`J6>]H;?&GA.&)7?&O`W0,^9_KY3$7WZ
MIUD1J/HYF>2S/8;:IW]JEDRAGY/)F%9QJU._R5T7V3U!KQP!R75BVB^OQW,2
MBE9QJY;YW):9V7'XH%T=AP]JVI8)TU)USILFHU+'!KYUC>BXC2MLA,AX!BO<
M9(+F<"MLA*^[PD[J<!"'P5*W&EAE<"MLA(9T2BMLA"I,RV1M\I?:F=+[P09I
M_Y_5M.2SD&UP[6MM\PI/]Z+>YFW>YDV+7L-D&CH``B5`!J5>)TU+/@O9!NF]
MUCO?$PI/]Z*^@5^7<*T+M0=2ZFF/C*A1--E)G:UA%;=:)R;@<&F_UCO?$PI/
M]Z*^@5^7<*T+M>Q%>()8`F10ZG6"C&>0WFN-C*C1SG)-&#YOY<J?Y71@'.XW
MZCIP:_!'5'"Y'*U7ZG42YT-*]H2WNF,/6>F^ULB(&NV<\`9&>*L[]BN<SC:_
```

```
M_Q!&]Z*>-070M.A%$E7IE_(E&YU<-"5`!J5>)WY9%[+A<"X07WA`]G0OZEE3
M`$V+7J_;&O&5![(!"0I0-"5`!J5>)WY9%[+A<"X07WE`]G0OZEE3`$V+7H3R
M<*CBEX.%![+!R"5`!J5>)WY9%[+A<"[08'A`]G0OZEE3`$V+7JN!50;7&@V6
M![(!"9A<`F10ZG7BEW4A&P[G`@V6!V1/]Z*>-06`<3\1MB121C_0&O&58K(Q
M$453`F10ZG7BEW4A&P[G`O&58F1/]Z*>-04P="-7=&3P`ZT17S96=+(AB"5`
M!J5>)WY9%[+A<"X07S96=&1/]Z*>-07@A=?!&@7NEZ$%&',@&X)8`F10ZG7B
MEW4A&P[G`AX&&'-`]G0OZEE3``N)2JPAEWZY>)TE&T53`F10ZG7BEW4A&P[G
M`O$E861/]Z*>-06``DE@$/;Q$V%+(NL(!W2G&%I-L9"Q5&,[!RE0ZF1/]Z)>
MZNF=_IQK\I?:F3\1MB2RCG`P]0:13='"\=J;S@[G`O&58OL/86MM\W,]`'0O
MZF!+CQ868W"@`VU>ZFD?6/>BG:UA%;<Z]63@`O$E877"RG/CRC_A6VG?N6/"
MF[YI:7Y)!BX07Q)6)RBP:3(*8++!!KZ5WNG^UY_E\VMM\W,]`'0OZACW+MY(
M(NL(!VU>ZFG?N6/"F[YI:2BPCS;LCS&R>)U5)RBP:3(*8+)QL&*@L'+@AFF_
M_NP_$?<^!_G^(K3%;<&#><WQ'`%I%<!"\$#/8&3@`O$EET6SCS;LCS&R>)UE
MT`[G`AX&&((Q$46SCS;LCS$26H`Q!Y"5_@G@A6`H!V+(;T7C<"[@88`Q!Y"E
ML'+@AG"XVI(#Y5X(AG(@AL<$672((B^+R0HK!VX(AU+R$8I=R_4^$1ZCNTR!
MX16/W$*EQ1"3EHZ*]VS1CFO@,4T!&/7/\=J;SA^^QM<B.7)-&.ZU90?Q%6$Q
M%F7!+@8F')!@`'Y)!B[@88!1<7:L4A^^QM<BB!`K`P:-T/^2S@[G`AX&&!57
M-"":`!^^QM<BB!!+`P:-T/^2S@[G`AX&&!57-/Z<`!^^QM<BB!"+`P:-T/^2
MS@[G`AX&&!7WR2KUX6M\+8((L:UHT`C]+^GL<"[@88!1<47#Z0GPX6M\+8((
ML3,@`P:-T/^2S@[G`AX&&!57-#:@4A^^QM<BB!`[20:-T/^2S@[G`AX&&!57
M-#>@4A^^QM<BB!`+(3A@T`C]+^GL<"[@88!1<46#`RKUX6M\+8((L4MDT`C]
M+^GL<"[@88!1<463`RKUX6M\+8((L34`(0:-T/^2S@[G`AX&&!5GQTZ9`!^^
MQM<BB!`K0S)``P:-T/^2S@[G`AX&&!5GQQN=`!^^QM<BB!`K`S!``SA@T`C]
M+^GL<"[@88!1<78,H@GPX6M\+8((L1:300:-T"WSX6M\+8((L1:307OVUQ"V
MUKVM,D[]X6M\+8)H,8@,87^]V!^^QM<BB'1TBU]:(K55-E9D><H"B=#39[P3
M[W_E\7_(\X3A\W0OZEX(AG(@AOQ6-"5`!J5>)UX(AG(@AL=$]G0OZ@HK!VX(
MAX)8`F10ZI(VMFX(AV1/]Z).ARCRLIA<`C!``WA0ZG5"ARBR!NF=[FMM\_L/
M870OZOLX(LO=36NA`UE3`"5`!J4^]9M,]G0OZA^.;"&NOCH``B6P?*5>)Q^^
MQM>2WNG/N29_J5A:4XIE!QAV/F5#(@AD&@3/\8E$*`^'*H+H<"[08'@0QP1G
M<(+H<"[08(]%&*];Q_]"!BX07WBPPNF,`B0>^P^'*A:!`B1.=EAE<"#@`_9C
M/8FV_Q!FY<J?Y71@'.XG`H4H&+%'+K%JU9DWI*^!(&U>ZF1/>*L[]I"5_K5<
M[Q,A'$,129!A\;U(8"!0B4SA,03/\=J;SC8/SSE?[0I/>*L[]HK-N29_J9T9
M:W5+!@9D_Y\U="-7=(+H<"X07S96=)`U<%AE<'5,=EAE<)#5<`^'*L=3-(3R
M<&CPV6O=F>S9FJUA%;?:Z3YOY<J?Y71@'.XWZCJ0*MT[6P=3E)DIO:5>)W$^
MI&1/>*L[]I"5_IQK\I<J'$,129#Q+;NBBW52B4SA,03/\=J;SC8/ST4$+@I/
M>*L[]HK-N29_J1@AO03/\=KK\7_(\P9&]Z).!!FA89!8ZNG]UQ#&]5H=FG^5
MSEU]\]&,`H3R<*C"`T4S`S+@E/L/88"5!Y/WV0F0!Y,GB)5D$:N!509GT&O-
M'Y,'`CY@/Z"4!;*!\R$`TW,]`'DP>8)X/[UDT`C]+^G,'Y,'`CY@/Z"$!;*!
M\S`]UP.0!Y,GB/<32@:=>=AJ'A5F;1=B'A4&Y7DP>;IO2*>$S`K_T9D)1BA@
M`@[W\N=!%'5"*`^'!E._R5T7V3WQ8"WSX6M\+75BVB^OQUPG>P2L%`Z_95VF
M.JPC9HN]UE@*%=SOG%)^5C.,`FH&!VQF!W#_G'R6&-&RV&N-I5#!_<XIY97?
M&GB>.P/A9Q"6[NG^UY_E\YF'K>918=9V(>9185"^&EAE<+IO2*>$S`K_T9D)
M1BA@`@[W\N=!%'5"*`^'!E._R5T7V3WQ8"WSX6M\+75BVB^OQUPGUXJ]95VF
M.JPC9HN]UE@*%=SOG%)^5C.,`FH&!VQF!W#_G'R6&-&RV&N-I5#!_<XIY97?
M&GB>.P/A9Q"6[NG/_USV$EZV.F#6.G^5SE@*%=SOG`I_5C.,`FH&!VQF!W#_
MG'R6&-'R7NF,I5#!_<ZI\)7?&GB>.P/A9V1->*N+`D3VG.FOIQ71IW^:%8':
MF>S9FMMCJ'WZIV;)%)W)GJW9&E9QJY%:OK=7D5+EJ;^CJPG`?I&3%YDIO3;#
MN9LFHW*5-0=0P@/T7(2Q:3(J5UES`"6<F9#5574A:$L"65U5%X*V),!-;BJ,
MP5+G!IVIV'\5Z3NO&#@?`C:O\%ROU5]/]Z*^@5^7F1<8!I!8Z@Y?R_4^$6]'
M)DS1F469A9EY@6'0'=H72>VH%$R1A=8B(B3B>4]4[Q,!&4L5?#KD,,!7@,,7
M+-,_)I%A><K"H4Z1NY-2B(+A>4\4(#-/&&H'1H)HI^%I%;>:WFM-4B95!*UA
M%;?:Z3YOY<J?Y71@'.XG`IW)GJVI`[<&?T0%E\O1>J5.]H2WN@T/6>D>>F2B
M=:L,$CP`Y6K70>G=>EHW$443>F2B=<0*W.E7!EHGB-KG&.EWB:%')EI'K#*@
```

```
MV+5<[Q,1JCY1?D2U,_CW+TQQ*HZ4A9ZG6(:8A9ZG6#8(*QQ_\C2G0SZ1A3_T
M!OUN><KR1ZN:@YZG6)EA-QEV%5EXJ^ZC0_4^$5GH>8IE@P"3A^:/6R^`A^VG
MAY-BB%GH>4]4[Q.1&7:385>1A1Y,'=81\-V!`J=B'5F(S`!B_Y^E=AUT[6B7
MA(P'AX*H=AT$68^G=9"`R7<N.5"N=AV4WAN()AV8-050-(^G=9S!VG_E\7_(
M\X3A\U:N_%E.!\;A?B*@=F`DB"4``S2`!Z4N:8P'AV1OY<J?Y71@'.XG`J%'
M)EI7-"4``S2`!Z4N:4I/=61OY<J?Y71@'.XG`JVG=1-1-"4``S2`!Z5>)ZU'
M=61OY<J?Y71@'.XG`H^G=1-1-"4``S2`!Z5>)X]'=61OY<J?Y71@'.XG`AN(
M)AW8L$53`C!``WA0ZG6R@6C2@>F=_K5<[Q.Q@5^7A9EY@6'0'0/'B>TX%)$T
M4Y`A2(&8;6;0',^1A;$WA7E0[Q/Q(EE(M[8VA01V%(9GJCW5',^1%SUA(['X
M0;D&%1CV(@%?[Q-Q5A>8%^-U(K_5*9N"*@TC&,C!/F+.*9"4!JR1>9WYF670
MF`$R\X0!!P-$-"!`=:`F:I"%$83A7EO69:K#.F(&86N-I5#!_<XYUP-P5C.,
M`FH&!VQF!W#_G'R6&-$"86N-I5#!_<XYUP-0^:V!Y[DS$'X&89QB):O<574!
M5;)AI^%I%;>:]IK<&CO?$W,]`%:N_%E.!\;A?B+0F2!0`G50L;%H!LWQ*Q59
MZG6R)&3O!ITIB)W9&E9QJW4"!P-$!G6R)&2_UF[0F3E?-$NB\!]MY<J?Y71@
M'.XG`EL>+)U9E)EY@6%0ZBD^`)7?&GAN8.L.!P.T'"LPXD$A&TORV9K<&A-!
MXM<Q0&I"GT$A&U0':J(F&SY@/Z6F/RW`/)P>T4YMY<J?Y71@'.XG`HQ1*?O%
M%+8*JR@@$+D&%2E0ZBD^`)7?&GAN8.F?`'W0*07"S$N2]FNM\W\8B#@?`C:O
M\%ROU5]/]Z).@.6?+:)"!@8$B:6>WFL=`C:__Q!&]Z)>ZF1/]Z+><`^'*I#`
M`$53`F10ZG7BEW4A&_MHP_X8(X.%!V1/]Z(^<%AE<%FS`$53`F10ZG7BEW4A
M&_MHP_X8(X.5!V1/]Z+NA2/&;XQ<`F10ZG7BEW4A&_MHP_X8(Z$%&'-`]G0O
MZ@N)2H$)]6-;<2!0`F10ZG7BEW4A&_MHP_X8(XO76>F=_K5<[Q-!@.6_JJ):
MD74R_7"21T7I<B-AX)G?$SZ1A;%'C`,4(.P_$4]0/\,%%]0E&'>1)7(QI)%A
MB"_R,CR3I?,?+(Q1*?O54Q4I&*MA,`$R\X2A=473F9]IH/=?(,RL=7'=>E2W
M_V2]_NP_$:$'-IY';8(1V0QT]$V1'5$"&!A&]"1B><KR8$![\M-R%9VYJCEX
M&T/Z'=`RXW-0!V+08\K2>8PW&Y@"]">OE^TW?`1$!@84($#+8.FL=>T,`J%'
M)E2G\!^M=J;(M$.?A(P'AS,]`(^G=8)XYY(#Y6K702D^`!N()ATHB(^G=9P!
MW!.1?F6@=46C?8Z1?G&XV%IW/$73F9]IH$X]!+W'>`U;-%0WTP/P!,NG=8)(
M=3,]`/\7@&`CB$/0>XSWS-C6@"@RUCE!F@6.`BBP:3*:(]EG%0%()F2_[G\-
M89ZZV%5\$B>?!`9QD7F!>F`S0..R&E?QD)%4&A<8!CH@)F%@-TX=]'.0RI\R
M!+W'>/#&M`#(>$Z]U0,P!+W'>()(=3,]`,_'>`U;-%27W3KAU$&?!`;1*R=B
M:\'"KR(;)>E7!G]G)_<!.TY]]3DX@U6$1OWQ78L=]`9#F$R1A9`224PE,JR!
MAXQ'!DZ]U2B?SD/0>XP'`CY0-.E7!E0WUP,@T00P!+W'>%G3`$7S!,M'=4J-
MA$J(A"U7%\C<!8+X?P%()C,]`.D_5TX=]%7`$<.'@X;H>8:H:F1@0'7"0Q>F
M+#CH$U]C%4X=]"MB9"Z2$1HW!+W'>#CH<06'@_L>0E5$$DY]]1D!&(*A.[I_
M(ISB>0"3A5\#,#SU1X'AU$%/<;HF(F:2Z"8X*5F8Z"#@8\H2(#FQU0.`_ZTQ
M!+W'>"#@`]C6@"BR!G,]`!)-`&N-A"T7&3[P$3-$`\T.V%:N_%E.!\;!Y1WM
MY8Q1*?O%%+8*JRB`A"U7%RE0ZJF.!ZN.Y](.UDBHA$C8<G6!S%T@B#ZX/GRT
M!4/0>XS'J2TS!+W'>%G3`$7C_"F$!ULP!+W'>)R:$^D_5TX=]*'J$Y`229X'
M,%GX-0!CB#-U(F;R*:I&!@;DU%L]`/\7@&`#"0U0-$/0>XSWS-C6@"@RUCF!
MA$J(A"U7%\C<!8+X?P%()MFM$TX=]$Y@'OI^'3V4A61PM[/Z(EF(A.WF<7!"
M*@&2$T&?!'2P<Z!R5SR'+LL!1T;P!'526\,19&J>;:?B2)Z2!TZ]U0.0>7XI
MB$C8<I'!&3W\+R#@`];V+\,LY:19X"B``ILFHSFR;DG(>%N0!EV0X@.`A"T7
M&7WLU%5\$B<O!%VU?#BX[R%416BT$2TP+2V`='G0'4X@4Z0L'DX=]+MB,,L!
M,%EXJP&2$UL]`,Z?0GBP!>E7!E37!8+X!,M'=3,]`#ZX/GRT!>E7!E37!8+X
M?P%()C,]`$^P?%H'"0I0-,_'>-FM$TX=]%"@FI'Q(ED(^6)2?["#@SYQ1B71
ML$E@$,/CU$$/?W)!0%F(;YS'%(^G=9:G+%//.?'_8TY]]3GH>8HE$"7T1W2P
M<[/!!IWI$1#2'4QK&U6_V$$O'G6"$79R'T>O`SCH$Y!O@GC8?L.G*$X=]*6!
MOZQ1&B`P!`P1!%+@U%L]`.E7!E2W[2B?SNE7!EKG`T7S>%0WUP,@T02PUFH'
M1E`.(<T.V&K70=M>`(^G=2I0-#(PTP6P@6C2@8)XYY(#Y6K70;+!&<!M)>D^
M$>N^[EI7-)WYF67P]>E.>*L[]E*5_GI:$7WZIUD1J)WYF66P/8;:IW^:%9WY
MF:^Y$I[Z.[KJ?TO!*5D(^29H!LWQ'%F8F1>8%^-U(FW0'2`0JCXQ0()Q+Z`"
M'+Y/$N]Q*F#W'%3*BI5D6G6"'5"A5761$3C0`J)5L?,J)AG1COSN$[#2(XKZ
M&'^)=S\1MO<%,%E(M[8VA1[Q*7DQI@LJ;69`;=T!`E?@$S2G0]'2(UDHK]IF
```

```
M!EEL+3*5@YE)*9FD<=\!%]\A*H4H&!J'`W5B<E&Q7\!Q*K)4L?.Z646A0SZ1
MA:*E;6:0Q=:2$=3%%C)``RT@6CPC$!`H)@52E;#2(S5=E%G8NGQK@B^2A4<]
M+JMQ%511@=GV]/!&PZL+&-UA,YS;;((60`,$6<TF:/+Z]-0&6<TF:/E7:RML
MA"[,N?+Z]%E<-`PH;6:0Q2\``CCPV0'D>592-%0':J*V!?+Z]-36!4=+[G@P
MOU=,=:`F:EL@KT^?Q<0:`UV0WFNM=F#D`T6#`PH?0)ZW`JL,`W@@0\]C$50'
M:J*V!?+Z]%E,K#+0!6F??U7)R`PH;6:0Q24``C@`60'D>2#@`SZPROE7:^G-
M@-)F!EF\`D6C=AWTV80GB'Y9%[*!K>KI>=&,`LT&5;*Q@6C2@6F_UCNO&#AO
M;M_<<`^'!B\`HH3A\U:N_%E.!\;!Y1`F`@PH;6;0`R6P?'4B!`Q4!M)F!CU0
M`LM7)_E7:SU0`LM7)U3Q@#U0`C!``X5%PSU0`F10Z@^V@-&V;0<GKT]/;2R0
M?[7&`AN()AW(`H27WNF?`(2WNON:(U*5_KY3$7WZIUD1J*1IFMMCJ'WZIUE!
MF@7NEP.!!SJR:3(J5U9!6%)USALY.%&U004*-C/#='61$70+WAS6574A:&A@
M>>\Y,TQ7%QD!DW-9R_6>TZM14YDWA>7A3$AG'@3/\8DT."^'`BA`XB#@;U>,
M`B0.`@WW<&@@&\2*`B3N;D.PPA[_AX&(\^;VS0WW<&CP`B!J8%:N_%E.!\;A
M?B)`FJ:I`^XV!"S@;X)8`F3``B5`!G4R."]7`LLW]0,!*$53`C.$!Z7^8"UC
M;G7B;W4R.`?7&EU5%U`E&U9!6.F]UH/#%#Y@/]:3:/L/85:N_%E.!\;A?B)`
MFJ:I`](K%1E!P$I1ZF1?^:V!YX21_K5<[Q/Q,E&2F91B*03/\=J;SOXF.45$
M<`:W_Q!&MX8V.'30!8)H%81ET&O=VRH3T4Z-!I8GB"@P$5U5%U`E&RA0CC*V
M!2C0574!52E@%835!5L``UW@C"`P`T[IU%E3`,3:&BC0574!5;*!`N4H8UN`
M`EU5%U"5`E9!6%UP9UW@C)N>`W,=B<=#K*V!`EU5%U`E&RA0CC*V!2C0574!
M52E@%835!5L@`UW@C(E/SB\P`_DZTRB?SFA@>2#@`T7TD14D)0GOU!)-`&A@
M>8)H1"OY0E+R$4IMY<J?Y71@'.XG`FWN*3:L'CJ0*Y8G%1E!P'9.FJ:9`J6>
MX@.P[NO^UQ#VT6A@>8*(`BC0574!5;*!`N4H8UN``EU5%U"5`E9!6%VP!3#0
M!7,=B<=#K*W1574!5;*!`N4H8UN``EU5%U"5`E9!6%UP9UTPUY%X/,3:&EU5
M%U`E&RA0CC*V!2C0574!52E@%835!5L@`UU`SB\`P0:V[C>)!I;7!8)XDVA@
M>5UPM!]AT&N]\XJ!\^;VS0WW<&CP`B`:T4YMY<J?Y71@'%S.V2+P,CIP:7C0
M`R50!RA0`C!``WA`)[EB>3U0`G6``B4``S2`!W121`;2`R6P?*7^8"V#`EU5
M%U"5`E:!8*W1574!52E@%0B6*Y9'2Y97)S>)!I;7!62?_MN#!I87-JO<574!
M5;*!`N4H8UN``EU5%U"5`E9!6%VP!3#0!61_DAV)!I;7!8)XDAV)!I;7!4?[
M$9]-Q)87-JO<574!5;*!`N4H8UN``EU5%U"5`E9!6%UP9UU`]B_9D6A@>5T@
MB"_9D6A@>5UPM!_QV41L>6&SREU5%U`E&RA0CC*V!2C0574!52E@%835!5L@
M`UU`]C;9D6A@>5T@B#;9D6A@>5UPM!\!6>E?R_4^$3Q)!@.=A5C0`HQ!;\]B
M%XZJ/"''%&_@3$AG'ARZ7V0PT--2[Q.1!2W`&/3V+'S"\=J;SLAL;M_<<`^'
M!OL/857,_A.1!`:1%_;A!K9W/DYG<C7,=2`P%)'D./&O;_X6)=-B]2</^?8"
M&3\Q/=V1!+9!!SNGX40G%V10)P019&H^+59_\DU7=+,A&*VK7J6Q'`"3A>5'
M5&/0'=#10]-B]2>/AXRW'-+Y$RHTJR^2A5HM'.W([W=Q`DQQ!@%O]2>/'9&4
MA4>-0X!A]0%F;E9I8&L=`D,W<D4WUP/@;]=^T(D1+1#VT10-&!8]<>8_UP/P
MT68-`^%N)?YVM$6#`TJ]UOX&`CY0-`.'50;7[(`]<>:_[07@;X)(`ZYN[F,-
MVHH1`^%N)?YVM$6#`TJ]UOX&`CY0-`.'50;7[(`]<>:_[07@;X(H`ZYN[F,-
MVHHA`^%N)?YVM)^LU&OM;R#@`T4S<%AE<,T.V!-G_MM>`/ZVVJYN[F,-VHHQ
M`^%N)?YVM$4C`X!^[CGQ[700[E9BY<J?Y71@'.XG`J1IFCJ@^>''%$S2DVY0
MZBD^`.N>[M.>_K3\.X?ZIUD1J)G90=MCJ'WZIUEQG<N1F6#DEV0P]006^P^'
M!E._R5T7V3U1)QA>%W5BVB^OQUPG>P2L%)':9'.6-0>@`@YGF9'U6#I)*`^'
M!I:IR>?+F9$=O#19DF7_%M9BF8(Z?>%KF0D`TY9YT!P!R5+EJ;^CJX-V,+-Q
M(>:Q<3B4A9DIO?!V$:H&&`2_N62=,W&AX0F0PXX!^YL=6'H<%S]1DHQ"%#!`
M:EC01F\41XMV&-U[24'C*)!R7U-X*8N5,W'Q$R7)*$01`Z2&!6WT1G'$J831
M574A:`9#!J2&!6WT1G&T:&FQ7U'2BDV1;<G+*8:82%U5%X+6*]U%:EC01F\4
M1XN6%F_@3,G)FXO5574A:"2D+*2&!6WT1G'$J0L/C29I'J2&!6WT1G'$J0G0
M:UES`"7\%(C\6;V6-0=0PD\!W.AIB7NY$2MLA'AP1RMLA'P$69LFHW*5-0=0
MPIFIV/S/92_A9:L#9JWS5^F,I5#!_<ZI\&<UPRB@9G#`9G8`]\_)9XD1+>^5
MSE@*%=SOG`I?^:V!Y[DS$'Y&UFO=U3<?S2A`*`^'*CQ0-#,@`TZY_Q"VUG[I
M$3Y0-/?32[*!\R$`TW,]`(1G>`:]UG[I$3Y0-/<32K*!\S`]UP-`>(8'6>G^
MUY_E\VOMEQ[A`T5S/[TD&S@?`C`]UP-`>(9GT&OMEQ[A`T5S/Z$D&S@/TW,]
M`(1G>)"5_IK<&EU]\]&,`H3R<*C"`T4S`S+@E/L/87AP1X)8219!*`^'!I"5
```

```
M[G_]63Z/!W<DB(3R<&A@T&N-!W<$`CY@/Z"$!7,]`'AP1X)X/Z$$6>E?R_4^
M$?<^!_G^(H>2$;.ZY<'"*(X"*=$[A7D0`O;_7NF<PXZQP_L/83GL&+`_R0F0
M>7X)*-;V+X`"Y7AP1QJM^WR$S`K_T8Q"%#"P!7S4!5;IU(Q"%#&P!7S4!59I
M8.F>[IGGEX!B;?\"*%".!W>DT;K/1\B\_Q!F2%O`1UU0QYF<_IQK\I<Z_<[$
M%`7>GB]2+)1B*3+R+P;1U>I1)^VH%$QQ(IQB2/4^$71`*!5?6ZD!%Y&3!AZA
M<3)`\!ROO>G<U3<?S2A`*`^'*CQ0-#,@`TY9SD%;<ZTIB"BP:3(*8+*!`BA`
MXB!`M[9&K"B``B3>&FF0!R_PHA:!`B0>^P^'!N0,67_-\592<ZTIB"BP:3(*
M8+*!`BA`XB!`M[9&K"B``B3^+P=.GT$A&X3R<&A`SI\MFVM!FZU1<ZU9)X3R
M<&@P]:#A`_$E89)&:3+6=8KR?<OD/I?G\+5<[Q,A'%W-^B`P`S(``W@@`_D:
M):2$!Q83^F)2(%5IFH*!$>=1[Q,!&<N!U;!S7.*1+:)BFO9:*&'@3.@EF&2@
M!D57%_6>RR2&83@X>.9!\!ROO>G<U3<?S2A`*`^'*CQ0-#,@`TZY_Q"6>7X)
M*$5#*`^'!O*K^WP$`CXPB/^"!\.L\!]M,&2P!3*@`GS4!5E3`$5C,&2P!7S$
MJ3EA,&2P!3*@`GPDOUUPJ:ML,&2P!7S4!4=;(F2P!7PDOUT@&R^P/#,]`+W2
M75L@`RK`1UUPJ473*]VU!7S$J3G1*]VU!3*@`GPDOUT@B!BZ3%O`1UUPM.^R
M3%O`1_+;!;+Q`LLSTP-`0LJR!3*@`GS4!77<%)>W!7S$J3E!0LJR!3*@`GPD
MOUW0L*M,0LJR!7S4!4?;%)>W!7PDOUT@&R^P/`:][@9#!EL@`RI`*`^'!O+;
M!9<ZR9BQ3%L@`RI`*`^'!O+;!59I8"2D+%L@`RI`*`^'!O+;!0T[R>E?R_4^
M$>&V'(;$H6>0$211:_1J-U=A2()Q!AEA(P5'!P3/\=J;SFG@$240Y83A\]@Q
M!A92-";`*$01`W?&J=N#'6,`$D5C`HQ"%#!P9YRZ/9GGEX!2-/*K^WPD.2"`
M!W>DT;K/1\@\UP.`'6,0`UO`1UU@E0:V3<1Q7T4C350`6>G^UY_E\]@Q!A92
M-";`*$0!`W?&J=N#'6,`$D5C`HQ"%#%P9YRZ/9GGEX!2-/*K^WPD.2"`!W>D
MT;K/1\@\UP.`'6,0`UO`1UU@E0:V3<1Q7T4C3N1$!9"5_K5<[Q.!3:O!%EEH
M2`3/\8E$GMG4&MM$'-XT]9M<)]@Q!J%_'4\1`W6"!W<T]:#A`QX&&)9E8`9#
M!M^W3.YS>2]_2,,!%W51)[WM%74BWNBCV+5<[Q,!B9!A=ZH&&`3/\8FDG^QI
M'JV1!@1V(>91)WAP1UT7V3U1)Z;]\GH\)XI=R_6>TZM14YEW^OM%!P3/\;C4
M_P;A9:L#9JWS5^F,I5#!_<ZI\&<UPRB@9G#`9G8`]\_)9XD1+>^5SE@*%=SO
MG`I?^:V!Y[DS$'Z67;3\.X?ZIUD1J+*Y%K2Y/8;:IW^:%;*Y%K39&L9D2'7B
M<W51)Q)6)V74=8KR?<OD/I>G(YLFHW*5-0=0PGQD2)9YI=15%Y9YI6,[!U*U
M:3(J=66TJ>DH55U5%W.6-0>@`@;S7)KF57*5-0>@`KW275+5574Q9UES`"I`
M0LHB59[Z.[I:]L!7%('8-)EY(>:AX1G1$_TUT,52125D,YQK,)E1M^3&1Y!E
M,)E1M^2FPGA<!IE1MYLFHW)%J7A<!IE1M^0V>2N<<A,-=XK-N2:/Q=P&*),'
M&#+R$[>QE,,4F`9AFAP*WX:8PP`/<E=A2/;_69-WJ47C<W4!6?#=QYQK\I<J
MEI`LO1PZ%)'T&.BR&#_&&J04W_7/\=J;SCX7&3Y@/Z"$!?L/89-WJ45S/Z%D
MT/"]RJQ,79'!&?<32K+Q`E$N)1]AT&L-WU#NE$X-WY.<_K5<[[G,1^TY_8^1
M\I>FQACV(NL(!]W!&'#P^JOQ'"]2+)1B*9C"\8F4>7X9E/\"*$7#O[H/)5`^
M>;J?!L@\]7B`S/L/89R,`$5C3(:T!7S$J9=)!EN0!EUPJ:O<574!5;+A&C*V
M!2C0574!52DP!EVP!3#0!63?*]VU!6G0!8*(`EU5%U`E&ZXA8UN``EU5%U"5
M`F/0!7?6!61/0LJR!6G0!0V[REU5%U`E&ZXA8UN``EU5%U"5`F/0!5L@`UV0
MWM.>_KY3$7WZIUD1J"F_F[WI6S*V/8;:IW^:%=>Y'"F_F[WI6Y;&RF/+H;X5
MJ48H85*U:3(J5UES`"K@6U)USILFHW*5-0?P,_>RPD:HPIQK\O;.(E+(F](K
M&/C&>70PXRSV+MY((F4T+K4D*"*D*['/F_6>TS+C$3!L%\ZT&,^Q(M:RJAG!
M+.7Y$V$;CVP!C[)A+3*U&O4>T"="(LC5CBQ&\!R?2)F'`M(IB+XU]59I88+A
M`X/X+\A<)Q(VS/L/8>4H8UN0!EVP!3#0!7XNG=<^=90F8UN0!EUP9UW@Y])Y
M[5-':3*V!6G0!5L@`UW@Y])Y[>F?_KY3$7WZIUD1J-DY!410J%E1$7WZIV9)
MYO="G*UA%;=:)W/0M^1F%;<J530*55ES`#_S7"OAJ;^CJPG0F3_!*W$19%K=
MNE#+7I4K?BL#'`E@6L`!`D,R(34``C0``E1:22!A(0'%0#XQ`L`AK`_PJ^>Z
M/4LK3Y?!J_NX3"#0<`^'*BVP`H/6FK'_<*@",*;9ETPP!<V9PXZ1L]NSM/[4
M(\+Z`+]ZKMNSM#,`'+RZC\M41`1G<"VP`H/6FF2'508'._'=ETPP!<V9PXZ1
ML]NSM`G5([PJK`7`J\):`"N0`/MBQW5RL"\VMC%F%*,*%'<@&$];D`<"K$VP
MM*Y543#`!Z:YM#5PI7*@T8*H611[;V-;(K55'=X$'+PJK`7`J\):`"N0`*:I
MT8*H62)6X`"C8<8Z$4X!%=-571LE20G@%%`Q75O1`PK+="X!*!^DU4X!%=.U
M%<`AK`^P`E9\/*P\MDOY0,MA!LWQ'%EXL"\VML!J(19B(19B(19B(3"PM#=@
MQ9!@`,AUAP4!,6D98T8A'/91/V6N&6F`M3TBK`_PJYM+UILFHY!I0)#5574A
```

```
M:`YG0)_M<'<+"0BPREU5%U`E&W8:GE9QJVX*_@YW*HLU!Z#A`XT%*,=3-`YW
MM\2*`BC0574!5;)AI^%I%;<J&SP`Y3B0]@YWMY"``*O<574!5;)AI^%I%;?J
MIC)(<`:W6',`&C[06!YQ/$7C<'=+K"B``EU5%U`E&W8:GE9QJ[+!`U".`P[_
M5V1PMY"``$5CI^%I%;>:WFOM<'<;S3B@4L<38MGVD)%4&C"W6H2WNF/O_01P
M&-TK!Y`XEX$%&CX@6"E6QZWA<'?K`SX``C<@&]$,`W@``SV\00&YCG#P`XDT
M!Z#A`Y/%%L>SR@YWM]$,`WAP`TX)`C[@`R!``^\)8[1E6[I-&',`&C[`7(`A
M&!.QR@YWM]%\`[)!K#UL7L:T=>X#&,DW!Z#A`Y,UMA7W$9(#Y7,`&C[`7(`Q
M!]ZO$W:B1W*92',`&C[`7,YUJ45CI^%I%;>*HM]O$4X!%=-573"F9)%%!G<+
M"0A0-'8:GE9QJ^F]U@YWMW'ME(I$5VMA)F7T`ZM%>*L[]I]->*N+`D3VG.GO
M.Q71IW^:%8'Z@,K)G(6:%171IW^:%=>Y'`^HG,S9&E9QJY%:OK=7D5+EJ;^C
MJ\BI5X(!?QN^A[D&%9I[&2`P)!-2`R!``R!`I94$$A824`SD$R,``,+Z`+]Z
MKA9B(2!A(19B(182`TL;`U8\$2<0`D#'&;&'LAN^ARVK:U$A;,UQ=W)!)3TB
MK`_PJR:;EYRC>XRW4<MCQ1.!LAN^AWG).;K'>.0>)<S#Z<`AK`]`K`LS$;]J
MLA7K$SZ7<M$*N+O:IP^P`MT*'":=`";-JYRQ@81I)I]"%6[P^F<O&)]"*G)A
M_F22!\":EYQS@0?$0/01K8#+,B:=`"8-N+O:IP]`K`LS$;_:LT1+M%9<N$1+
MM,`AK`_PJ^=J(19B(19B(19B(3`0K8>2$?*Z62[7&N?CN`V4:U!QN7WZ`+^Z
MN63=574Q9SY7%T=\8F=`)DZA%%NP1)S*N6MM\_L/870OZE.0G$8!?QN^A[D&
M%9!8Z@Y?$2$P$=M3^:TAOXZ92.DOK<S8&E9QJRCZ_1:A%#VEE[K-N?C?\M05
M&:O<574!5;)AI^%I%;=*SIBE%$Q!!G>;2#Z_UB1E4D70&E9QJYUNSDYMY<J?
MY71@'.XG`@^HG,RI`[<&?T0%E\O1>J6>X@-0^:V!YP:6_I?Y=7YY>Q7Y?63B
M%$I1)SY7%VFOR:U!4B95!*UA%;?:Z>;LU%:N_%E.!\;A?B+P@,K)G#IP:_!'
M5'"Y'*U7ZBD^`)7?&GAN8.D^[>GO.Q71IW^:%8&:G=19J%E1$7WZIV9)YO?R
M!=39&E9QJU/?M^1F%;<:<4T&573C<%+EJ;^CJPG0F=+[P>Y:N>*W,L!QR:8%
M'"`P)!-2`R!``R!`I94$$A824`SD$R,``,+Z`+]ZKA9B(2!A(2`!$A8"`TL;
M`PEP`BP`=)S1M.C5$[K\N&S!5")C&L`AK`_PJ^>Z/4LK`PGP=F3"%!^4<`!C
MFH)A!LWQ'-,"'+S:L$V+7@PN)2N0`"11E4/SQ%FHU2C`!%/0G#GL&#F[/4L[
M`\`AK`_PJ^>Z/4N;4#W"JPW;M.A%!>\O)2N0`&]')DSQ00D',*8I&&;0',\Q
M+<"Z/4M;`PGPNCB4A5J-`DPP!<V9PXYQN7WZ`+]ZKMNSM#8`'+S*M!/9<`^'
M*BVP`@G0<`^'*A^4A0D',*;9ETPP!<V9PXZ1L]NSM#<`',+Z`+]ZKMNSM#@`
M'+S*M!,Y<%AE<"VP`K-(<`:'0UF8<`!CFGW)!%/0G#GL&#F[/4N;`\#!J[S*
MJ[S*J\):`"N0`$W0`Q;"&1CW+MY((NL(!U.?5*@+SJ9Y`L#:!'R0!'P``WP`
M`WP``WQ@FA8E0PG0!#T`$IP!MO1H83%F%*,*%'?`H2P&SJ9Y`L#!J[S*J[S*
MJ\):`"N0`$G0`Q;"&4V+7KAGAEIU%5/`K\I2BD$&F%32(U:2!#T`$IS1M.B%
M>V:H55<Q="-7=,L!F%32(U9BFAHM)9G5%(`A&/T5WQ98B'_49#;#N9LFHW)%
M!@:DV+5<[Q.=G$8!?QN^A[D&%?;_61A!&#Y/!G>[RILFHSE23\S8&E9QJVFO
MR:U!4B95!*UA%;?:Z0&MNG0\]@:]UF1PM^T,SCL`T8M->*O;\`:]UF1PM^T,
M<W@@`QOMU`^HG,S9&E9QJVG?!YU2(,Q,!G<;US"`!S(P!BAJ)6EQ`BP`=(9(
MG6]%)@"&765_MXR,`ILFHSE23\S8&E9QJVZ:%B11E8Q!;P$,&CX07X!B)45#
M!G=+K"B``ILFHSE23\S8&E9QJ[+!`U".`VE/!G?+R"BP:3*:(_7$C*UA%;?J
MIC%E%(Q!;P$,&CX07QYA)45#!G=+K"B``ILFHSE23\S8&E9QJ[+!`U".`VE/
M!G?+R"BP:3*:(_7$C*UA%;?JIN#O<*<2P*#A`PT&*!-1-&1PM\2*`BBP:3*:
M(_7$C*UA%;<J&SP`Y3B0]F1PMXR,`ILFHSE23\S8&E9QJVXJ@P1G<`$,&C[0
M8!XQ$45#!G=+K"B``ILFHSE23\S8&E9QJ[+!`U".`VE/!G?;R:N\:3*:(_7$
MC*UA%;?JIFF!C&>@`P$,&CX07RE6QZU!!G?K`SX``C<@&]$,`W@``SV\07=)
M(NL(!S\0P*#A`_%E8T6WRF1PMS[@`Y(D&]$,`W@``SU\6&P\<D5'!C\0P*#A
M`QX&&((Q$:M,!G<;S3<@&\0*W`[7`CX07W)IQY(#Y0[7`C[@88`Q!Y]->*N+
M`OR=_KY3$7WZIUD1J-FI+X6:%171IW^:%=>Y'-FI+ZUA%;=:)Q)6)[X5J49H
M%;=JF5<ZMG-0J2>6-0>@`KXE59[Z.[J:`)TY&Q868Y6>L3>KM@/2(Y=L6L`!
M`D,R(34``C0``E1:22!A(0'%0#XQ`L`AK`_PJR9K,,N!W$*U41\Q:`1D><IB
M@3!F%00!*!?3(\+Z`+]ZKKTRD,@M5$LK`]*Z3*6L+!8(8U9!$(!R,3TBK`_P
MJR9+0LJBX2JK+4L[`PG0LLI2RLIB@3!F%00!*!?3(\+Z`+]JL@:S',@M5!L%
M`C0P:`1D><IB@3!F%00!*/)T&<+Z`+]ZKKTRD,@M5$M;`]*Z3*6L+!8(8U9!
M$(`B3Y<AK`_PJR9+0LJBX2JK+4NK02VK+*6L+!8(8U9!$(`B3Y<AK`_PJSUB
```

```
MT@E@TB@P:=2O+(*1>?%?%4,O*A*V,C;#N;N5-0?P,_>RPD:HPIR[UCO?$PIO
MY<J?Y71@'.XG`FW>F1868W``B:6>]IF'`M(IB+XU]8(H8;H/)3XPB/\RS'4B
MG4=;-/Y,&#[_,_<BB,?#RCTC&W8:GE9QJV2O`B@@G1HM&ZO,5+)AI^%I%;=*
M]BJ``M*)WH+(RCTC&W8:GE9QJVFOR:VQ\SVA\!]MY<J?Y71@'.XG`B\C!SJ0
M-050`C"``VP`*"4``SC`!H!2`C"``VR`!Z7^8"V#`B3>&M)))V"-`B3>&M*I
MT70"UBA`XJTAG>B=X@-@Y<J?Y71@'.XG`I87YL=3`C,$*"4P0X!2`C.$!Z7^
M8"V#`EU5%U`E&^HLG70"UBC0574!5;)QT=*IT>3\8"V#`EU5%U`E&Q<MG>@]
M$0:=[NFOR:UA\_L/870OZEE3``I+1I:6>26P5&-[96U>ZG4B862?>2@@G8+H
M6U,OB!*F^U#B`X/X+\-<)])YM$7CSTY-]Z*>-050`C,$*"4P0X!2`C.$!Z7^
M8"V#`EU5%U`E&^HLG70"UBC0574!5;)QT=*IT>3\8"V#`EU5%U`E&Q<MG>@]
M$09-]Z)>ZND][>GO.Q71IW^:%8'JG4$`GMMCJ'WZIV9)YO?R!=^9!.%I%;=:
M)[K<$Y%:OK=7D94Z-R2LR\&[$I[Z.[J:`)W),V0B&&;0'+\"J]5R>Q7I+9!A
M5F_P!B,8+.-5E"*@RSWA+6;0'&T`'/C&>3T%J_45(AO1'82WNA8B'/818W3`
M*8Z4PY%$>7.0>TM2M$IVI7-#PM*YPBF'8`;F6)^=>2@@G8*HR\&+$M*I^]*)
MS/L/86MMI^%I%;<J&W'MEW7AC-(YUP-`>*L[]I"5[H2WNBA`9,^9_K3,.$1$
M$VV$!47`!"A06.B3`'*P!2B`!RGP`CC0!15EQY(#Y2AP`RV``BB`!RE0`CC`
MT>E4/W;:1EA0!$R``H6%/@D@!UN``GB0`B^``UWPS*T!$CP`Y2AP`RV``BB`
M!RE0`CB0`ALO97WZIUD1J.29386:%171IW]JECV%3:NAF=M$'-XT]9OLJ&,0
M^K,Q!C%0)Z=D3.X#&!QJ,&3P?<OD/I?W\H<T''!1%W2W+KY=)]]M&I%JA-M$
M'-[4D&Y@TMM$'/?U0=+9&M*TH,$B3N1$!72JR>=[6&Y@TLJMX2J[;)O,7JPN
MO_GJH`:3&75[5@V;J6-PBYGH!B:-;YS'%.W)*(X"*6:RY2&L``:3&75[5@V;
MJ6/0PQL$A28-^1=B'K,1-(X"*6:RY2%LI:=D3.CK!B9-$N]Q*CA4B()A2'5B
MFIY%;A]V6&Y@TG?4%("!8?%-K\-B87#F57*5-0>@`@;S7(?E!B9M,,N!W$(5
M:W4+9UXE5UES`"K0*]V%OFY@TKTRD,@M5+%6MW#F57*5-0>@`B2D+.CK!B9-
M0LJBX2J[;)G);-!HQ6.L`H;TGE!HTJ]A'[%6MT($I&]A+>CK!B:-X16/W$*E
MF.OBVPWI!B;-8K7MVS]!%`##K[F-6S\@B]]M&NCK!CK0L/JINV]I)C`V2*3B
M69[Z.[J:`$CW!I\1&G9!&J:!&MBT&JID$'F1F9NE;(Y1E!_4$_TUT,521254
MM"RS3<1Q7X98YIJ1!HTD&(S:&M*TH,$B3N1$!0`R&T'C*)!B)EN^%L`QA9="
M*3Z1A;157V@X$LOV05E(CR0A!]WQ%!9BB*W*(;SI$4\!$F;0'&T`'%DH?'!!
M78*!'>;?'8F^E'GQ&O81:[/R\H<T&X(1`RMP2L;T`CC`,V0B(S;#N0:3&74+
M7WCPGOC&>4Q1X(ED,)E1MX`%!W@`60:3&74KN]W[6GA<!IE1MR@<XG,`60:3
M&74KNS(`65U5%X(F!]3Q8#EQ!M3Q8#DA!M0AQ&>P/64`W!O)!ILLQ&>P/<OG
M9A@L=</$0/31Q$1!3!\DKQ@6:W4[Q61M\EC,;?ME+R.RW-U46=TA')O,7H9(
M*F<0*793)Q0-&)QR2JYZ(/4^$9!A=W?A',_"7BAP*M15</_"%$QBNOC+&J4A
M)JD,F*ID$+@Q](5'$.7<\:WQW)(CUX3A\VRPR8+8/`:]?`TMB"A`XE+RW)"5
M[G_]63[/!ILLB"A`XO^RR0:]?`TMB,T#6>E?R_6>TTI_(>;1&O>>O'91SHLU
M3`P4BW9\M'9"%,;T`C@`69GGEX!B;?\"*%`^3`Q$'[K/1\BL\(:T!7S4!599
M\K8<]-(Y&VEI&F4#)Z)B(P6GZ*IR%88T!R6N/"11E1B+*X8T!^7<\:VQ3<1Q
M7^V\:D4P3?L/857,_A-]+X]4:P0/]+;,_A-Q(IQ2X`!C2`0/]`&6>7X)*$5S
M2L8DO[K/1R#@`];V+W@PS`K_T55\$B>O%/87XO:5!K&8>:99)\J]'/$[OTX]
M((L=]'=2!CJ`@SXA,B);V[Z-VY$R*>(1[B=ARR?!,F,),"_E`C*0`SD``E)P
MM###G`C5`K?5TM###G-S%``P'EU%L]`!0-&!;]W:8QUP/PT68=40YETM\/0`2T
M$3..V=0AB`9#!EO`1YS:,F=`'59YW=1AE=?MWZ"M&/:$3_KT42:-6;TRD!LQ
MXYA-'58)UF=`'8+8*]VU!7S$J<%-'59YW?X-VHHA4`3E4";]_19!0LHBXP>^
MVP0@!]1AE6!]!M1AE==-'8)(0LJR!7S$J1)^[J"M&!&E419E3_BD3Q]ETI"]
M*,K3*P.)V=0AB`9#!EO`1YS:,F=`'8+8*]VU!7S$J<%-'59YW?X-VHH141IE
M40)%4`YETI"]*,I#0LK2X7)`'8)H,&2P!7S$J2US!M1AE==-'8)(0LJR!7S$
MJ1)^[J"M&/:$3_KT41LE4`3E4"8=7,L$;TUARIA-'58)UF=`'8+8*]VU!7S$
MJ<%-'8)(0LJR!7S$J1)^[CGQ[71@TM]O$:YR'9!"*LGWTM0AB`9#!EO`1YS:
M,F=`'8+8*]VU!7S$J<%-'8)(0LJR!7S$J1)^[CF1[A.QUBRFE7,]`!)-`.DK
MB"B``B3>&JTA!]1QQ3/@E&"=-0=`K)"P`!A*'5=<`SD@Y5ES`,0*"0N``F)`
M'5<<`QL-`L[X$YLL&R]`GV30T!8QORD^`.D^$7^]V!)-`.DKB"B``B3>&A="
M'6"=-0=`K)"P`&=``'6"=-0=`K)"P`&)``2#@C#^QR;+Q`C]!!F[FU.GOU%5\
```

```
M$B>/X16/W'""JS#7`Y,-@3"'!S<P$3V0O'RRV`.RV%>O*TN9A:M!)DSQ05EH
M!A,S(F>@`Q0Y$$Y]]3EH><J"0QA>\4;&X7YY%86(*8L=]&9M(<%R`Q,A&\G+
M*63I+'*RJCFXY6L1(#DA&141`A.1$W``*$7#1T?[$6^VVP,`!X!2-'R4W0-0
M!I54-.E[M+,Q!C"P!7"`!YR:$V4`$D53!I7$&1A>%XKMU%5\$B>?O)RRY4LY
M!U3!FZ;Q`TZ]U2COY&+0EV6PT9(S&V+0EV4`HG,]`!)-`%7\/">?O';1'7S7
M[R,2\`3/V1V^Y<%2-&4`W!.1[A/QUXLMT010Q<]S\I.]!MT!8X!!\)S=X5L>
M+$53!C*0W01`$VV$!47`!*W!1R]_!RD^`.D_5TX=](SB*)`2O5L>+`!#*EB%
M!H!X802_V`.RV$$O,B*;?2TP!19\/N%^$K9\$BPS]W@@OT$+!X!B)7"`!QKM
MU`.RV)<AK`8@K`;PJXMMM#<@(1OE;N^V43S0IP2P/4]Q,8M]&<)J`,)J`,)J
M`+^ZV$8+)*VX434@(1LU`Q(2L^P^!O*4$Y<AK`8@K`8@K`;PJXN]U0/P%#"P
M!7"`!_+;!4>[RD-B$5N^%K+A`SX``C1P;5['FQ[Q'0W=BEL]`$\1`UL`!WB@
MT5UPM*M,I1:QY6LA&S[@`R!``S,]`$\1`UL`!X`"`EUPM*M<`U>\Y6LA&S[@
M`R!``S,]`$\1`UL`!W@@OUUPM*L<M%N^%K+A`SX``C1@T.F>[G_]63Y?Q>P_
MT?=R(5AE<`0/]+;,_A-Q(IQ2X`!C2`0/]`&6>7X)*-;V+X`"Y:=D3+K/1\BL
M\!]-T8!AT=]M&G,]`!]MUA'E4";]_0!$0!LQXYA-'8)H,&2P!7S$J2US!M1A
ME==-'59YW?X-VHIA3_BD3Q]ETIC5*P.Y$3..V=1AE6!]!M0AB+W275O`1YP:
MW-1AE=?MWZ"M&`)%4`YETM]O$22D+#)^X+M-`')`'58)UF=`'59YW=0AB"2D
M+%O`1YPJX><.VHH141IE4?:$3_KT428-V8NB/+TRD)A-'8)H,&2P!7S$J2US
M!M0AB+W275O`1YP:W-1AE=?MWZ"M&!&E418E4`3E4"8-V8NB/"2D+!TN!]0A
MB`9#!EO`1YS:,F=`'59YW=0AB"2D+%O`1YPJX><.VHIA3_BD3Q^U40)%4`YE
MTL&U3/#6%*:,V=1AE6!]!M0AB+W275O`1YP:W-0AB"2D+%O`1YPJX>>>$]].
M!R;]_1;A*M<!*:22?"]-'8)H,&2P!7S$J2US!M0AB+W275O`1YP:W-0AB"2D
M+%O`1YPJX>>>$^D^$6O-8EHYUP,@T020OH*(`BA`XJW1&G)`'5<\`TX)UEES
M`,0*"0N`H=1QQ360`U*>-0=`K)"P`"@@!M1QQ3&PT2#@C#^QR;+Q`O1)!@UM
M$?.;X@.0[A/QUXLMT020OH*(`BA`XJUQ(=0!UEES`,0*"0MP!M0!UEES`,0*
M"0L@!M0!`L[X$YLL&R_P$V3@9DZ=_A41`A.1$W``*$7#1T?[$6^VVP,`!X!2
M-'R4W0-0!I54-.E[M+,Q!C"P!7"`!YR:$V4`$D53!I7$&1A>%]F-\DXN!GU9
M!ALM.;,A!GU9!B`ZUP,@T02PY<%2-&4`W!.1[A/QUXLMT02PY<%2-&4@`]E-
M`#311EA0!$S0&GST\G>0X@.0_G/EU$$O,B*;?2TP!19\/N%^$K9\$BPS]W@@
MOT$+!X!B)7"`!QKMU`.RV)>1-0<@K`8@K`;PJXM]&5)B)3ZP4>[V;AMU`Q(2
ML^P^!A>SV)<AK`8@K`8@K`;PJXMMM#,@(1M5`Q*R44#2BC'+[F,@3SEQ&<)J
M`,)J`,)J`+^ZV%L]`$\!`UL`!WB@T5UPM*L\)!:QY6LA&S[@`R!``]?F=;SI
M$=_1T*VXU0/P%#&P!7"`!_+;!4>[RE1J$5N^%K+A`SX``C0PTP/P%#&P!7``
M*"#0!4>[RC5PQ5N^%K+A`SX``C0PTP/P%#&P!7"`!QK=!4>[RD&[Y6LA&S[@
M`R!``P:=[NF?`(2WNF,O5>GO.Q71IW^:%8&JG^QI'MMCJ'WZIUFAG^QI'JV1
M!@1V(>91)Z=D3%T7V3U1)Z;]\GH\)Z$X$6GP6"MA,)E1MYLFHTPQQBI@2%)E
M,)E1M^1V2L9DF0$=V<$;BA-AVI89T-`;AYWJ.W5Y&?HY&\<%&[$V*]W!8H)1
MH"`0!#;#N?!W/E>QD04:!A;R]?!W/AEFA`4:!B#Q]?!W/AEFA`4:!E0:FH21
M,W'Q$R69`$%0FC-EBW",62]2H"T0!&]%7MTTEPE@,)E1MYLFHW*%!Z(%60:3
M&76[:3(J5W@@KY!E,)E1M^3&1XK-_USV$EZV.F#6.G^5SE@*%=SOG`I_5C.,
M`FH&!VQF!W#_G'R6&-'R7NF,I5#!_<ZI\)7?&GB>.P/A9^IO\I=*]3^Q4/]B
M$(ST:03/\=H;Z4%0FC.E\$%0FC,EB`4:!A82FIQK\I>*>I!Q$:J&`O9Q(=#+
M=:E"!V-0SATO[)E]\_&LVC@?`C"]_Q"6>7X)*-;V+X`"Y:=D3+K/1\BL\!]=
MQ2=Q\CD<24KAOPSD$]/_&"`B*H4H&*9)\(N]U0.`!_(JB(:T!7ST`CC`J4Y=
MQ2=Q\O#34\%26WU>YB)"(CIMKYN"*OE'7<LG)D2E*$X=]/L58V[@$1K6EZ<2
MB&I>?_9F$*[2%#>V!DX=],'B*IUBV<4K71['NP""@X8X!V%@-X+Q0U#HU$$O
M0R70'7?B$Z?B2`4:!C30&K5T'-,C&X986T,*PU6_V$'OOG7K&*<:(#FQU2B?
MSG@@K]\,`WB0KRKE1,EI%,:K(16K@!ZMUH"R`D5S`S,]`.D^$0C]+^F,!_+Z
MS3"`!T9@!"KE1)SE<$UQ8U'AT@,@T010H&%`I2@0!*4Y4U-?6/]"8/^"!RMP
M`U-/8$P)<WA@!%92)^7NW("R`D5S`S,]`-.^93#GU`C]+^F,!_+ZS3"`!T;@
ME+@X(GYQ8U'AT@,@T010H&%`I2@0!*4Y4U-?6/]"8/^"!RLP`U-/8$P)<WA@
M!%92)^7NW("R`D5S`S,]`.D^$0C]+^F,!_+ZS3"`!S!@!"HE@R/B%S<6%2X]
M`!)-``4:!E2*`D%0FC,U]7BP`C0P]03V+WBP`C<P]03&E#"'!T9@)752[LX-
```

```
M*"M0-#<PTP,P[7PV$'ZVV`AM7JZB$=F4?!X-UG@@6@UKQY(#Y2AP`RW@EWA0
M`CA`SDJ]UG@@K\\,`G@@6G-=``4:!B"!`D%0FC,U]87U+P26\#,]`.NN709&
M>(8'6>E?R_4^$4,123!]M`UB_^\5Z9E]\_&LVC@/T]_<(/L/86M]2L8$`CY@
M/Z"4!7,]`*=D3()X/[UDT&N=!A[A`_8#2E@PUP-`>(9GT)GGEX!B;?\"*%!^
M2L:D^WR$S`K_T7@@KX)H2%O`1R^``YR:$VN-!_+ZS3"`!_FJ4DZ4G$9AO!I2
ML0KHT6H-*"M0-#<PTP.0[A.!T/^2SG@@K]\,`WA@!$:@4D[$60[7%#<6%2X]
M`!)-``4:!E2*`D%0FC,U]01F)T1A3$^"!_([]01F)T1A3$^"!RN``W42^C"'
M!T9@)752[LX-*"M0-#<PTP.0[A/QUXLMT02`!Z+5L'8L.5".`C?0`GZ)!R6`
M`^2LU&N-!_+ZS""`!Z(UUP50H&$`$B@0!*4Y4U-/8'9"%,;T)'@@OW"?X@.P
M[MIE8(1G>)"5_K5<[Q,Q%)$$TU+2(/;_7I&>V3<?SZJ-\S#]S2B]_Q"VUJ=D
M3"#@`_8#2EDPUP-P2L8DB/?32P:]UFG@$3Y@/Z"$!7,]`(1G>`:=>7X)*-;V
M+X`"Y:=D3+K/1\BL\!^-!_(JB(:T!7ST`CC`J3FQUG@@K]\,`WB0KRKE1,EI
M%,:K(16K@!ZMUH"R`D5S`S,]`.D^$0C]+^F,!_+ZS3"`!T9@!"KE1)SE<$UQ
M8U'AT@,@T010H&%`I2@0!*4Y4W5R/Z'T)'D@OU-?6/8#2ECP)'D@OU./!RMP
M`W42^C"'!T9@)752[LX-*"M0-#<PTP.0[A/QUXLMT02`!Z+5L'8L.5".`C?0
M`GZ)!R6``^2LU&N-!_+ZS""`!Z(UUP50H&$`$B@0!*4Y4W5R/Z'T)'D@OU-?
M6`D_TP.P[MIE8(1G>)"5_K5<[Q,Q%)&4V='"\=J;SIE]\_&LVC@?`C"]_Q"6
M>7X)*-;V+X`"Y:=D3+K/1\BL\!^-!_(JB(:T!7ST`CC`J3FQUG@@K]\,`WA@
M!$:@4DZ4G$916\AQA0R4_[S9*98])RX]`!)-`'QTM$5S`S,]`.D^$0C]+^F,
M!_+ZS3"`!_FJ4E_*FYUBV?G/FTUQ8VOPZ!X-U@4:!E2*`D%0FC,U]87U+P3V
M+WBP`C<P]03&E#"'!T9@)752[LX-*"M0-#<PTP.0[A.!T.;E*AJ13<GGT6"-
M!Z+5L'8L.5".`C?0`GZ)!R6``^2LU&L=`BB`!_+ZS""`!Z+5Z<=3H&$`$B@0
M!*4Y4U-?6/]"8`D_TP.P[MIE8(1G>)"5_K5<[Q,Q%)%DVO;_7I&>V3<?S:J-
M\R$`T_L/89GGEX!B;?\"*%!^2L:D^WR$S`K_T7@@KX)H2%O`1R^``YR:$VN-
M!_+ZS3"`!_FJ4DZ4G$9AO!I2L0KHT6H-*"M0-#<PTP.0[A.!T/^2SG@@K]\,
M`WA@!$:@4D[$60[7%#<6%2X]`!)-``4:!E2*`D%0FC,U]87U+P3V+WBP`C<P
M]03&E#"'!T9@)752[LX-*"M0-#<PTP.0[A/QUXLMT02`!Z+5L'8L.5".`C?0
M`GZ)!R6``^2LU&N-!_+ZS""`!Z(UUP50H&$`$B@0!*4Y4W5R2L;T)'@@OU-/
M8`D_TP.P[MIE8(1G>)"5_K5<[Q.AVI"Q')D=+1ROO>F<V3<?S:J-\R$`T_L/
M89GGEX!B;?\"*%!^2L:D^WR$S`K_T7@@KX)H2%O`1R^``YR:$VN-!_+ZS3"`
M!T9@!"KE1,EI%+6%'%?(0`KHT6H-*"M0-#<PTP.0[A.!T/^2SG@@K]\,`WB0
MKRKUI;S9*9:M@!X-U@4:!E2*`D%0FC,U]87U+P3V+WBP`C<P]03&E#"'!T9@
M)752[LX-*"M0-#<PTP.0[A.!T.;E*AJ13<GGT6"-!Z+5L'8L.5".`C?0`GZ)
M!R6``^2LU&L=`BB`!_+ZS""`!Z+5Z<=3H&$`$B@0!*4Y4W5R2L;T)'@@OU-/
M8`D_TP.P[MIE8(1G>)"5_K5<[Q/A*AJ137U)?S!/\!ROO>D,T]_<(/L/86M]
M2L8$`CY@/Z"4!7,]`*=D3()X/[UDT&N=!A[A`_8#2E@PUP-`>(9GT)GGEX!B
M;?\"*%!^2L:D^WR$S`K_T7@@KX)H2%O`1R^``YR:$VMMVG,]`!)-`'@@6@UK
MQY(#Y2AP`RW@EWA0`CA`SDJ]UJV1V3<?SRB`!_+ZS""`!Z+5Z5:RS9>*`B&0
MV3<?S2B`!_+ZS""`!Z)%SB`PUP50H&$`$B@0!*4Y4U-/8/]26`D_TP.0[A/Q
MUXLMT02`!Z+5L'8L.5".`C?0`GZ)!R6``^2LU&O=&IE]\_&,`G@@K\\,`G@@
M6IUN)=M\J2@0`IE]\]&,`G@@K\\,`G@@6N0,`G-=``4:!B"!`D%0FC,U]01F
M)T1A3$^"!_(+]RD^`.N>[NF.T/^2SC#]S2B]_Q"VUJ=D3"#@`_8#2EDPUP-P
M2L8DB/?32P:]UFG@$3Y@/Z"$!7,]`(1G>`:=>7X)*-;V+X`"Y:=D3+K/1\BL
M\!^-!_(JB(:T!7ST`CC`J3FQUJ8]UP,@T02`!Z+5L'8L.5".`C?0`GZ)!R6`
M`^2LU&O=&IE]\_&,`G@@K\\,`G@@6IUN)=M\J2@0`IE]\]&,`G@@K\\,`G@@
M6N0,`G-=``4:!B"!`D%0FC-5)_<32D^2!_);)Z=D3$^"!_(+]RD^`.D^$7^]
MV!)-`'@@6@UKQY(#Y2AP`RW@EWA0`CA`SDJ]UJV1V3<?SRB`!_+ZS""`!Z+5
MZ5:RS9>*`B&0V3<?S2B`!_+ZS""`!Z)%SB`PUP50H&$`$B@0!*4Y4W5R/Z'T
M)'D@OU-?6`D_TP.P[NF>[H1G>%*5_GI:$7WZIUD1J/KY!1FS/8;:IW]JEDRA
MGU\`H#KBJ;^CJV6O:KO>!C;#N?#U8-NC!D)\!D1\1T)\!@GPGW?T8-M3<'?T
M8-LC!FAP1RN<<G@0`P\&`MN#!S+P8""P/7D0`P\&`MN3!S(@52#0N7*%!I:W
MPBF'!VU06`:&!VW@!BN<<GG0!H5E8'G0!FX@Q&>0`'@P!WJ@V+5<[Q-O%<MQ
M1Z6L+!CV+;MB@>RK'@3/\8F$!G?4L)/\GW=DE811<'=DE8F,!G=DE821>7YI
```

```
ME?\B.47TD16D^VF`S/L/86M]DVG0!0WK`[ER1RI%Q'?4L$5SDVG`J>C<&B?9
MD6G0!2#@`Q>"!G>D4@GPGW<DB"?9D6G`J>C<&B_9D6G0!2#@`^^"!G>D4IB!
M!G<DB"_9D6G`J>C,+S>9!ET``C[0%&AP1RJ5`&*`!G<DB#;9D6G`J0F0_K5<
M[Q/Q!L[D*M^R*[I(\!R?2'@`$G:<KU*RZ3$'`@F0!R!1-,TC52"0`'A0246#
M!R!!K#*0K[J?`'E02463!R!!K$L$69$)H#!9)T:TDB]$2W=4)W@0`W62!S%0
M)W@@`W62!S)0)S0`]W#O\+5<[P%-']R&!G<D)AGA*A="F]^R*[HH&&22NI`"
M)_U.^<.B$=E4[]>*!I9G@7G)FYI_%:4A)E3!FZQQ*I"1*Y8G_F[@$5;Q'7!Q
M=T$6\.7<\:WQGW<$`CZ0*W>D4FAP1X+XGW>TPNE<<'<$`CZ0*W>D4FAP1X)8
M<'>TPNDL!FAP1R#@`[ER1RJ%!G<DB&*`!G>DV+5<[P%-0+O>!@3/\8F$!R!1
M-'@`$L0J`S"``["%!Y54-'@`$L0ZOY`5F0!ZD@92)T:TDB]$2W=4)W@0`W62
M!S%0)W@@`W62!S)0)S0`]W#O\+5<[Q/1*P.YZVU`\!R?2'@`$D6#!R!!K#*$
M`["%!Y54-'@`$L0ZOY`5F0#ZD@92)T:TDB]$2W=4)W@0`W62!S%0)W@@`W62
M!S)0)S0`]W#O\+5<[Q-!0LKR+;NBBP3/\8F$!R!1-'@`$L0J0S@`6WA0246#
M!R!!K/,+69$)H#9I('5B1"OY0K1T1W6"!S%0)WD0`W6"!S)0)WD@`W5"`W`/
M]\^9_KY3$7WZIUD1J)&9,86:%171IW]JEDP1F0"*!HY!2XYA0[1T1W6"!VW0
M]WG0!GV/!VU06`;>!H4EU-SB?G8@2$7G/F3R1#IR5I>J`FC@&):)!HYA0RWY
MPW=DF7C0!FY@F7G0!FY@F7C0!GA@F7G0!GA@F8#(+:\5BG8@2$5GF4YQ81TD
M59[Z.[I:]JJ6%[O^6\=B&CH39&&`!^)!2[:D!QNQFZ6L+#("$TSP`DP@!9ME
MPW<#'&6N&6E`0"I"8O/&+0`BU-P2F#HT-#Z1A8QAPW?S+WY!;VSA`R]5M"SC
M3&2.'68N*GYY%:8Y![*1&79S%1IG!X)4=-TA'&AP1TOI`TQI+R,2+3V2A23Q
M'J<2)84H&):G+'5R'ZLA!X2B&!"3OY%"CS.NW%<-&$6KJU;\!9#P8$;[_7V*
M//"]&@:C`ZV!!VW`3'G0!FZ@4EE3`'S``\,LU-S2/A<10&Y@K$>+!9"0`5D#
M`19E)<(!B-S"%C[P4L;J;IV<-2$`N%O;L%@`"2C0L("[M0V+!8B3-1<`N"PS
MN,>3-050M"PSN,>3-050M"PSN,>3-050M%O;L%@`"0V0-1T`N%O;L%@`"1F0
M-0Y`K''[!9#``H"KJPE`*`^'*A]T(ISB%&R1&79S%1KG%!<&1GYY%:8Y!P!R
M+()4=&)22X^R'(XC.\#A%!?60;D*'%@`"18@)9"@J^4&"1:0$TYQ81V$`CW@
MS\"!!9!@`5(""1P0M'8@2$6``CU``]P*'%@`"5(""2$`'%@`"2``'%@`"2!@
M,YP+7RN<<FJPPBF'!S'P8""P/7@@`P\&`MN3!S'P8""P/7D@`U(%`ITK5VA@
M>4)\!@F`!W.@!XI=R_6>TZM14V1@0`3/\=J;SG8@2$4G.0EO8'8@2$6WVBN<
MSBA@!X)4=%>,!HYA0[+A`ZV1!VT`*)R1!VUP%<2ZT0IO!X)4=(+('VT`*)R1
M!VW@!K+Q`KGB&#:TPNEL!X)4=)*3\`9&>(:WPNGL%!<&1CYP+()4=`KO%!=F
MBL<B2$6GV+5<[Q-O%<MQ1Z6L+/]B$*0B%^9/)GE`\!ROO>F,!G<D.7)-&#Z/
M!G=DE0:6>7YIE?\B.;GB&#:D^VF`S`K_T6N-!BF9!ET``CZ0*W<DY6AP1X*(
M!BF9!IRJ[H21_K5<[Q.AGT(-*&1"7<JSZVU@_^^5SH#(+6SA`R^U_Q`VH`7:
M&GC0!F[0`H#(+>Z7!VW@!BT`B-SB?GC0!GBP`H#(+>Z7!VW@!BT`B-R"/@8]
MH`7:&GC0!F[0`H#(+>Z7!VW@!BT`B-SB?GC0!F[0`H#(+>Z7!VV`!RL`B-R"
M/@8]H`7:&GC0!GBP`H#(+>Z7!VV`!RL`B-SB?GC0!GBP`H#(+>Z7!VW@!BT`
MB-R"/@8]H`7:&GC0!GBP`H#(+>Z7!VV`!RL`B-SB?GC0!F[0`H#(+>Z7!VV`
M!RL`B-R"/I"5_K5<[Q.AGSCH$P4J(W9R'SYQ*JUO-$[AEM/C+3ZW?+/1DV!3
M&JW"8I94[Q.!!I'T8S82BQ\4^CCH$P6ZE$,127F!$62>$>V($79R'Q>A:L%3
M$O;_67@P!WH@B"A`XJV!!VT`*)R!!VW@!NF=>7YIE?\B.;GB&#:D^VF`S/L/
M87@`$D6#!VW@!@:=!R!1-'G0!E=!K'YYQ78@2$5']GA0246#!R!!K'Y9%[*!
M`BB`!W.@!U>,!BF9!ET@&R^0*W=$]GE02463!\"MR:V!!Y7$&7@`$CX@UQ'M
MU)F'`FI@E2"@!I+3%&3R1+JO!LA<)WD0`\A<)WD@`\@LY0-:H*V!!S%0)WD0
M`W6"!S)0)WD0X0F0[M.>_KY3$7WZIUD1J`-:H(6:%171IW^:%0-:H*V!!S%0
M)WD0`W6"!S)0)WD`HJ&(!S%0)WD0`W6"!S)0)WD@`U+EJ;^CJV6O:GE1H&#W
M'"B`!S'``GFPT5&"`G@@`RR0!R#:'2PF&`4*`D%@,YP+?^=S%;U6H&%@(5\/
M?^>382E7H&%`I:%)&#D3%S]1DD%0FC-EB[.L_B;?_@L%.]F6PXXQ]0;!2)]F
M_^\5Z4%0FC.E\$%0FC,EB`4:!A82FH11H&%`I2@0!*4Y4W6"!S%0)WD0`W6"
M!S)0)WD@`W42^C"'!T9@)752[BN1_GI:$7WZIUD1J(9?J%E1$7WZIUEA^"PJ
M'FA($I&:<G"1+4`D59[Z.[J:`#5\%8;_2B_P]#;#N6L-%]D"1`H_!--_!WTI
M`EPP`S-0!L)_T(D1+1`V!--_!WTI`EPP`S-@!L*?_GI:$7WZIUD1J,2_%MMC
MJ'WZIUE!_&+O\A<"\Z$H1I9I2%+EJ;^CJX+Z!OUN^%%B2"P@1N3>E$L9RYME
```

```
MPZ8!(!9(]^IA,YPKP$!CBW@0`T*PPBDG2-`-<W@@`XC\6;NU26=P9S#`J80A
M2,J?SIMT!OA<`F.0!24P!B4P!G!3)ZS<,[)!'V-0)ZS<,[*!K>9QM&("!V-`
M)RW?,[+!RIQUM&("!V-`SI`U!--_!WVY26?PG.FOIQ71IW^:%8'*]32Z/8;:
MIW^:%5ROU8VY$I[Z.[IZI4J?P'U?2WUN^#;#N4,P_7?0ER+`!3,P`T4@_.FO
MIQ71IW^:%8%J]D-:J%E1$7WZIUEQ5O(!56??&AD;J2F7L=Y/`#K6`U3/`C'`
M8W#'`C+0`[K3&H27*I""N]O:5[Y3EY>!]:*2EYSC33)"=IS8CK#/$T4I`_E:
M1'J@I*+Q'#M,)9M+UCD3%UQU5G/&OY"5,W'!56<U9[L"!Y^]D7,0!G9C_G``
M6<TF:+=;NS6TN_4VEX1!T8!AT1F[_Y]EUC!@TI!@`)"=Q8^C:U&A6Z$=`R8-
M"0;@1$#].+H6%0$V!V%@-^9O%$7C@_U%[DQ`]O`]$46C`BAP5B@A&ZR.!S3@
M%&3O@_W5&G,0!G9C_G!`]MC-N68M`R8-"0;@1*\+;ZD"*;B;9/TW!V%@-^9O
M%$7C@_U%[DQ`]KMB%%92-"J``F>%$K+!ZGA``TY!]C[87ZTQ!V%@-^8/!VFO
MR:VQ*T8!Y?P[UP-PN[D[XD$A&RBP*T;!&?!-K#"`!S=@!&/,!++Q`LN3K[,,
M87^]V+>;NR,>%+*!`KMB%)S!O[+Q`LN3K[-,U@E`>*N+`BA`XJD"*;B[K9"5
M_@E`>*O+ZDS@\.GO.Q71IW^:%8'ZI,HKI3.P/8;:IW^J943D!$.`!%7@!$OP
M2[ZS54^JO%)*I2CEQ#J"Q"I@!DXL5>=L,)E1MY:*$G"@/I!E,)E1MY:*$K@K
MNGA<!IE1M_#U8#FA!BMLA!R,P5*7!I9WQ">6`.JS!4XP!$A0!4ZP!!H-`YR:
M`!L)!TX1O(2QD7,0!G8C2)!E4G*4-0?P,S-U(OI;\K8<]#)S5UGH`E$\\#+?
M\:V!`N-U(DBM%D2%:ZUA!DZ,SW<P^++1SJUA4D6`$J\<RTRP_Q!&]Z)>!-VK
M2D3E!D,O*F#N`J7^8#EA!DY,]H2WN@T/6>G.98)XJ_;1&N-U(FW@\+5<[Q.A
M%'BR>?JFU2_S]\UL_D3A$T5?5JEB-VX0\[_,\8D$!TZAV="!!I?B%"F2WKC+
MCX+(RG/CRM?A%,%;\K8<]$^JO%!LR:L!BW)`\!ROO98GB#+@_00`V;55'?>%
M`I4DKW/PZ$^JO*W!975B`FE@>77"WW]E>;:(!S<P$=Y/`&F!'7L2O2@P`Y4D
MKW/PZ$^JO*W!975B`FE@>77"WW]E>59)&)GGEU;Y+Y+S$36@^VF`S`K_I,K;
M&EQ6)R:0!I97)_S-N29_J4^JO#@X?^KJ!@3/\8F4>7YIE?\B.2`P`S+@/!3A
M!$.`!%7@!$N@^VF`S"KE1.I:HS,@`\X3B?(Z!X_N\W"@/I=:-.ICT',0!G8C
M2(+H@_U%[DQ`]IF``FI@E2"@!I)3VT.`!%7@!$N@^VJ`S'.=Q.J#S'Z.N_QX
M[0G@@_W5&G,0!G8C2&F?`$^JO*W!9>YS9G7B!$.`!%7@!$N0WNF?`&;P.(&(
M`N-U(FV0]H2WNF,O5>GO.Q71IW^:%8&*I5#!_=MCJ'WZIV:))ZL!%=SOG"OA
MJ;^CJU>*+E`A&)DW_FP*;YW!$ZLA&%P*B&1RA3]H&D6K9)K<&D(P_3G3&B"J
M4M]O$<(Q!$_@!%/P!`P!0H*1%TP%B8OE\_C?&D(P_3G3&B"J4IC5F3P#-OB/
M@\1#*0HX$4(P_0"#`GT=6SVSJAG1F:V!=#R!!@YJ8(2WN@WO_3IAX/.*@\33
M*9\R_FR*_IQ+>*L[]MY/`(N4$=$O1.FOIQ71IW^:%8&J_=R_/8;:IW^:%=K/
M_<ZY$I[Z.[J:`-HO&)DW_FP*;YW!$ZLA&%P*B&1RA3]H&D6K9`F`_\(N!-.?
M,ZT!HBKU_181_IDW_FP*B:OUV?C?&D(P_3G3&B"J4M]O$>A"Y"6")D6)_SA(
M/*LE!-,/,"C0U^9%)CX1_3NE0^<O1.E/RWO:IW^:%8%JIT-0J%E1$7WZIV;)
M%'8:GEP6J2G'P2MQSILFHU*G/G<&`YS:N5+GPIQ+>(+8F:W!9>YS9G"?]FM-
M>)(CUP8&C*O]V82WNBC@EW61/F>V!3#0!<^9_BL!&@A-1TE&+DU!3@`````#_
MK@L``"<60&G?2`<4```,+KB(``$$Q8@4(-J<26,&1`P7,4#DB/&"X@T0<M[,
M*>-FC9PP#10$'.@F3)LR+#:.83$&31@X=,K(86%'YIPT;]RP(!,F)@LS;^2T
MZ0EB#IT\;,J$=$$%"8@F1Y(8<0A"Q$"K50<^<<(DB9,B2Z<X=1*D"5@%"AF"
MX-("!)DT<^"P"9,'1!@04:>F&7JF#(@T;A+F?-,RX\FP3J=D<?($RI0D4Y8*
M3)"VX180+>8$[@+B<@LRFSMC]LO9LQP0I3&W02VZ!9[0GO-LECP04@'/9F!C
M=L+Z\HH<,$#PP0R\-^8W(,RD2<I:.7,7D`;0IDUY8<,6:5JC:;[<+W3I(H4D
MM!YRRU+Q;.=P,9/$3<C2Y]=^7M_^?>?X;)4J@!]>OAS[YO7'5AL`XN<:?>[M
M=Y^`+>2!X'M+B?&?`@&Z@%X+9CRH8(47.E%@?[_!$!(?!@+WH87RO;%4$LEU
M!Z%(24C1XG/1+157?)49F$9Y!J+QH@LQSNA=C2*)!0(114PQA!1)0$%%$EO%
MUP1Y"KP5UUQYS`'"&&^P$11>4OW%5QE:YF278&X0AH9A?DU!A5MPR467"R$E
M00<(=6RDI1%>YD%&"U/$!)@8,IT!)QUHR`28H70@=U,;=<P5TY9=!D6G`GD)
MJ>488<@A1UV(^E4&'C&Y<5-.6G*A`@ILQ.&"=:JB($<<=KE!AEUR^,5E&W#D
M.L=&M^:Y:$A,:'$%I620>2F*4S(44AAL>'F'EGF\40<(C?[EQABYAK$1"%\F
MRVT9WOH5*K@U?11M51*F<08:=+A!YAP#O='0N7N%T5=(*&QD;J(&SA&2&7.=
M@9"W(-Q1QKH(DX%<&&[4M:L8@/6$4V#V:IONG?W>\1((8LPUQAJUWGH'&FE,
M"D<:>"P\1PHLA)01'3WY%2*XI[507+)G^#H'"PFEX>F7K)9AQIW9RN$NO#"'
```

```
M]"5'<\G15U%#K9MO7UJB,"H<$-\J1EUWF1'&&(V>9F](,F@+\AN()IP&&6UW
M#4*B2].1PK)00"$EE=#.@9Q<;&?]9:^`T?'RF2@;]4;/)H';4.$<W41'&LK6
MV5":=PKK!J,`,[BC`F_`=#$(*``61QU"YP$SMIVCF*,"@!O.NE^)-\KX:H!A
M>P=R0`G5TQPZ.(30N69$RL;<<-G^41M`WY'R=E\ZCZBU=(1TKA-4*)EP&74O
M"K3<,B!4^^++:]D[".2V1*F7,]G%AL=9XNF&PG670<:E[8WK[:+:EBHYY5E+
M@PO*X`(0L*%^I[$#M.I`IH/E"@2QB\FM'':'P)QK?+>[%!4212E>:21EM$L>
M^1J7)S+5RB[ITI=?(&>JE-4%!5_"GO;H]RX)(D0,Y2*#TRS(P5R1`6@]*P-'
MOF<KD+&!@2!0X!%-2)2[K*QE;-!@XA)"+E.QCBA/=)G;VB8U,3B.=#!@01B-
M8`2$.(]A<(`#N4XCO>V<:PXFT0\+)0<V+661#5IJ(T;.X,6,H8",8A0C0H@V
M1B,($FAEJ(D;IE>'=\UN(S(K0Z_(Q!&DL0U:;*C+6\Q@!IE4<GU!T5+N(*6^
M-\8Q872QBY;N$K*QD4QN)P-A2."`!K:-D)8)LQ8;;O42-78*:(QTI`R'L+WN
M;>XO6I*7_>R7G"_Y[21B4N$<EF6DQCQI*Y&!$12JLAXAL"4-ZY&"".KDAM,)
MS2]A$`/U9B<DD-7A3@X#`>;@=*54INQ23W@G'-Y)J46&`3"B=$/OAC(Y,QUP
M#2L<4U'2H`>4R+,.;2"4V1I",=FI\31WC)D"SL4E]B7D)3\K64A.5H90F>TT
M\\270N'2/YF(C)G.+,/I*ID&:"VK2=PT@S=;@(9PCE,!0?#2,?6(KW*B[E/?
MP^/?,D*&.HP!G1!4VB*9B<'E76H*:AP#0_(P+`;Y2`%]XUU0GBJP\,CHFR"X
MZ39%T,WT(,BG(<%J&;1J!E!Q<`Z<8H[8R/:E;/4RD]A"CDI5>"DK+-"$O5N1
M/$/"J<!\9'-^,8-AJ)*M&,``!E(,X;MD\M"(VN2C"&7GU?QRP!IR-I;,D19G
MW0!1B6II**$=[+[`FL8UNJ53")5#9MUBM#!$ZDY*1")+:8#9%:VUK?-ACQO@
MJ@"YTM6N?DF6'=+PU):V\"@:S,/*\IK$PYHO*(I-4&-!%MG)RH`&@06!#&`0
MG4L)09.]_2TPIQC<%6HI6<J1%R^U-`,0J(!T$>$!#T"6LI==J@K?0MEF4=K:
MSYYOM'E\GF`>6%$[<O:.I"/@&0KX&<M&IYD3J@'.0&`#`AL.:)]9[X?/=Y$O
MX<#$!@Z)$-CV512Q18=@M57`9IDK.APE(4[=SMAV)1>XN/&NIT34HM2:TYV6
M@;E%P`/7BCB'JB7E-!!.KW*0=M=Q<42*+&5I3@![OAFL%P1X4.]E2>=,*\L$
M(1!F,EMURA8Y,%<*B;2)7UHYLI(E#&4J8YG+Y(S<-C"W"4++");'1&@Z'TBY
MS`U4IR9W3"]QJJ`8:TA0W%6QXXT6:&GF4E#>4I*8@#G"ZQH*'%0)9/5E[%R;
M7DA)/*U0HTR:?T11LAW?4#CK";;+N?JRY=A)TP>.MD6R,],;O1P8A+$4AJ>!
M6J?Z0H;5Z;J[2T1F2-R5)A\V>J<.@K09Q!E7FLF!TH;"<,9BW>EH]@5H$GO#
MJ"MF:N,VF2T9$C>Y%6"$Y<3D-!T-95&RRA#*>2T/X1TIRM3'4HB!X`DRREA$
M9,5,.<EK=6E#01`Y`L&Y7)R(MT(O"D*&1(N7X6XA,<+3\&`2N3@T6V3XR!W`
MY0;`GLN':UE5J^ZHGE7-"F@E+,H;3J+DS87DH'Z)5:OP'9',Q&"]L9K5I9S`
MMG\116%%J66DO"9OHXAVJM>MRZLY6.R$JA!D8$O.V,KFM'NM28AJNS:G-@*T
M1&WK7W[!*[0BNW9+V7O..W4"<Q&,]RW-Y5?4!<$PT7<ZBYGI?$1^)_^.V`;`
MD.FIIBKOT$$@A2.(I[X_LYY+[@07'=@(#DN10C8I#X(>@`"S,LA!#C@/@A6\
MW@4UP,%%CE#[V\<@!N@50NK/XH(;*>`))G7>M\X%>ESYI0[SXYYI[Q>2#8I9
M=&8:<UU@:\*-9![$FMKB=@)^&E4'],_4/;)^*ODI\*=O_)4Z30GORY!.YFJ1
MV&;@=T_C0R`&6R=^1G($%!)4-RF(DFL<M'9U`"V,MX"8AB?`@DRF)Q+&YP*J
M=W00U7JW-P/!(2.VAUDU,'N\]X$0$1'")Q)20'S&%TL,MRE>\BT<82WO\FTA
M<F>7-"G'MC-EL'&;8BU3)0<M,`:6M@;*-X#:=&\Z`P,V2#,XJ%`ZR(.&1QA$
M"!=&""3'Y6AO$$YKP1[.\63C]E/Y1`?[="=W`05)T%]=N"689#\K(B-IV&A)
MT(7,]03R,F)M$!1^D2E=&%+!)&3K<D;'$W2F!$W-ES%ZIU=]-R%R(UW4M4)@
M1T?X0WI:0BA>!W/(<8</=(`84X=IJ'U_T1#O]T6R94+S-SLA`4?0Q%J>-5$P
M!D$7)FA1A!@@8`1)P`1)LA0"%1(J\"IJ81N9<FQ[N!3*(8LB,`5%4`0@$`1,
M,`5/\%/6@0(UD`(1,B%"4`=2PUER<P3+<4!D`C3M\6]KEP8U`0)#,'1CR%E'
M\!&T1%U9\Q8FM2C26"1.001)$`1'P!AND@1#D$V&I336DD>=X@:+LDH^5@:\
M@F[I!1==4C.4XBEU`!/,M(>[E2<+>#PG\2O29!?JQ$\RH6AEHD9N()#'M(>(
M1`=C0$UC\014<(L*D`76HH:!,2HIXQ;6R#^CZ!8RQS]P$!0/B`(LZ$9A@%!:
M<BY"A35V$Q)?@SX_B&L1`P((E0<H"0)"4`5'D$T;Y$E9ITM>XQ=I(D]"="OG
M$RH&@F.A@VEXHS<*``40(Y*&DC%1,S59A@(N($#Z86.N\2PZ9E;R41?QR%+S
M1&1)46_RF(Q5T!00AXMF(&/;R'D:P1$>`3%Z(#.-V1$?`01<X@8U@0<NL"O"
M^#DN8"164`2J!R4>@A;6X1`0(1$481$8,9F/N11%@`4A`1H(4D5!1$U%`$X`
M`````_VL)```G%L=S<<=@#P``#`T4M#F3Q@R(&"YB!&P!"4``@%30I)D#8B*(
M,"#<E+D#PDX9.7/2O'$#XHU!C$>2&.F89N-'$&;>R`'19N0;$%.HS'$!(@D=
```

```
MF&GPE*&(<<Y'D77F!!13Y\Q%-V0NDB%#<<Z;-F4R;H19)@R=.G*&\G2R->9,
M.PACZ'BH($$+$$/0A'%S)FN+-"#HW&PAAZU;N'+IVHV3=V\:OV^GT`DCATZ9
MJ'JE1FU!&$6<.G/II-%3)@6+C&]^CKD*ATT9QXA!&`D*@JG3.VGHH`'!6<Z;
M$X\OCAE3ITT=-EY%DF0L)TP>BBC2D"03&\U+VW4T:TP1,`$($'<DCIE])PQ%
M,664.R4S,BMLV2#*>)23!\28,&S8%`91?(YC.9NSOD\J'@2<-S)5=UUR+I3!
MDW*CR1'6&'2DH%H+4X#`G&Q'T95:$%.!()%];YQ17!LPR91>&-O1=X88'<%7
M1U8HE,%&&FTHYU56%((@@G*.N1$2'7D(6)(<9'PD@H83Z>5A&&U\IM<;U"E0
MW5NKX0'"'.^Y\<4038"`@FL5D70&@&:T%P0;8^0QT@M'%-$$$TD(05]79#3Y
M)`A-A+%&5D&:$<9O/[5`D4D@T`!#:D@4V>&'H'$$(QPD.A:5&.VUX(263E`Q
MQ1!RMO46AI,9]`8<FHT4XDQFI,'&??V-QH:((I0VUU`BL!6111:AI!)++I$J
M8DUNW)333CW]E-10JJV:!QD0.J8<>'*,UUR%3ND54%APA&64&Z*]L>I,,(91
MUY\DB9%&"Z9Y)%^0=J0QAEATMA>6&1^5X<:ZV'D'@A!@U3737&0$=(2I+P[U
MF0A)8/M1HVEX!!=IT;UT1'%P:#>'"$JB\493L]4(A1!-E):4?R2NX6U6_`:4
M1&LNMJ2P&6*`")]5&H:A<$@Q`K<O&U_B)UL;R&6G[FPGQU0'5!>9<5]/V-F&
MFD#MO7&'&RY`,L`;W+G(1M0#I,#3K!35"H(+=-`!!U"F/165174QEIM9(`P+
MDVTN!S2'<_%I^>*=()SQ&Y):@^#$334-^]EI8VSM7)>7I<%>26ZPT1Z\7H'%
M(F5F@]`"&DW.?3$;4?W7V'Q5N!%4Q1>?D;%S85V4.@KDU2$&':;-,8>#X(7]
MTI)L4"3;7'EM&)#,89@:AABFN2#KAFT;1=&+&H$0Q&<PQX8=P/[9)L;PCH/V
MTQTRK<'X12!00866Y/5>1GNC836];`'%T(*X/S$WAZO'/95'=WG\T/?)<BE,
M7O-64=^G-!,C/01G))]ARD^D1QYB]8H.T^I*?-HC`BJ%!4DBX(E/\F(G8@WO
M8C^I00VF5(;10$5W-RD?1F8@`QC@H84P:)M&\,`HJ#PF(-VJBP9CIK`'YF4H
MCJK(]IIC/Q#83%\5:</(D).&`O$D!C"0`0WP`$4I.D@VI4,##G^B`LW]A@PJ
MP$[W0*"#%O2M"6^PCY9F\`(:.`A0-2)@5BPR/WDMD"0UFA#JQ/,[HLF/?J,"
M@9GJD)<[=.5.(#%<5IB7E2#0!X3*\6#JU)"4G\`A.OX)BHO,%A`R%(<CC(3>
M3:3G1<[Y1R8_B4R-Q*?!$T!F<;%)(";Y)[.L]"I\5#">`H+@!C,U[SVIBXP8
M9&(;C@A0(S^X3ADSM<%21B4P9"@;?.23PZR$A#-SD*5H7@0'%-(D#%("E`T$
MA0=!P4"#/QF)XP("S=BU;6R1P<'[I%=-8)VL?#Z4C7*\1Z'4>4V*\_Q)2E9B
M%B5"4`%SN9]Q=&DRT?!.:3/Z3(P4))//*&<]=/@,O])#PXW6DR=%Z*B.A%.1
M[P!Q?5I4``M=",,D+E%+/BM1ZXA'HYN,TX7F]$SOLBC$Z=6-9J5!'S>U=#VC
M]$L!HFH!'H9#M!;DP0T6-4BUAF+'*]ZD5*D\G`7+("^>-($@!JG217)WD^W,
MI2[FBR!P-*,P',EK1TTS2(VJJ27EF$]"S\(/7?PC%Z-\QFU^(HF(6D`&-V@-
M(JB;(U%(HIS$I6Z`)(T,)=58+1QU*287&28FYZI$M*+@FF5(8&RR&1#"]<V1
M&S(2H@`5EK4F+"MF",MEY#6&EOS)(&X=J6:(52NR5N\-9*C#NLB@R[\YIG=>
MX6@-0R(J.!Z.KK7:#6E>-+?<0&I*$C%:?[I2HOK<)R`H*&9)I*K)W%W1.20I
M57SZ8]<:6<R=UP6J--?CK?[4*`QD4`.)[!@0.)07!%(X@IOLH")VD84CWOT(
M45*'WR!%9;(_N2ZUNB*=:!T..&HL9D!B.AN">(0B%[2*&["7%6OQR:3[,<IX
M[VH$"*6'HC.1'QWPHT"2$LA`6Y30&QP8&B,"R'MWH]'A_"N4W*F.9`S"S`1!
M((-,%IDB$C'=12@R2)JDP70'+0,-2]@@ABJ`:R75'OC@\!%X,6A*QW%,DEJ#
MR092A'MR\)XAZ\8VK/!$"$-)0Y"R*<B+\;`,`:F15<"RKL_(90S>"P/^%!DS
M(('L<]W1W>&"E+ZIRHZD"5TT6Y[0/.<*63P4.1GWOI@>CV!+R=D+\GS*5ZL,
MI?902$K/9>"SF0.ZH8^=2V-(B(<^;8EHUMBJ=:B@NE.,(6T.=8##?U1<E[#9
M%T`PF4N9Q/C%@"@1;]*K2>K@C![I2;:2I_S<QCK&AJ2@D]JF?-E-P),1==54
MQQ=9RFF.IH8WH"BY5*+U7N=J,-W:]MSJCIFD22:N%_%H/G?0&<G\I:87$*$,
M9_#.7>;V-?_.@&RPA=M=M]K5GLRKW`[NTEV7O4"B3&E1I@E(NG(52!=8'./V
M_+.82R6ZZD)FR',1W5[9=L27U),%2]FL7'[B3)2!K[7!,5=[NC*']AS:>Y_E
MJHWT0IXAJ;<S7GX"&CX3-)E\B0XZN"M8)51"TEB+N22)M!$+$L387@4$X_88
M<L"J%V6WH4D<%OE7Y)!V>[W(:&O3>*7/CH(8]"TB)3ZI161R91E=+5AA+NH-
MD4J2-BR=,0&;"<E7[-\8Z(576F+;T`8_%*-6/`TQ<.FW]C<O/=O1<4!7`"T5
M5A/FE"HWN_M)XHZ+$?B=?%Y9.1G)<R.]KL@A\[VS2!Z,SQ->'G6@89XQ?LN@
MQ#BO>`A)F,)G0L<:KTD_2-7W'J""(+[!T:%PX4->M6`-HHD$)+92%WWC8!2;
MW`R+T7$O-T48];GU7X_78>9#*(!%QF8?`/-^80%;(M(K;M`"Z8,5<E!;\)$\
M6`<";"$O]98'6,,6"U%P\F$;U[(&Q>$&>C`M:20O(A@&0&!"'H$'+I`^`0$:
M"%1/1$\`10!.`````/\@`0``D16@.(*T>0$```QNRI0A`X+.&Q!FTKA),P>-
```

```
MC@8*("8XB/"-'!!LPL@Y4P9$FC9A.,YA`4($G#!N1!1$T]%.FC)WX%BD`T+C
MFSINR#R,J"!!@A9XW"3``V+,&XL$WYCQ")(C")!PX`PL>!`%#!8P4H!XXP;$
MG#%R!+J1^#./T#Q%C\I)NO1CR(Y0I1(T",(J5JU<O8(5"Q%B&IIW;K(A**;C
MPC$=Z9(Y2(=E139LWMQ1>&:G1(IAPQ`,>.=I&#@(Y;QI@S`-FS*708"PHS'-
MS3EIW=AYPZ8.'==N8*-,*E5.F-M<8:,HX^*,"Q!H&!H\XYOTG#9'&U-F(1$$
MI`$)V=`I(V<DB#)TQJ1(C?'-F31CPD!&VX:[4[<BZP9Q0H3D$RDDL=PG"5X\
#1!H`
`
end


From 5151815432331149508
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,913afa6429982d91,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1991-02-01 17:00:46 PST
Newsgroups: comp.sources.atari.st
Path: utzoo!utdoe!david
From: david@doe.utoronto.ca (David Megginson)
Subject: Missing file for dvi2tty
Message-ID: <1991Feb2.010046.16683@doe.utoronto.ca>
Sender: david@doe.utoronto.ca (David Megginson)
Organization: Dictionary of Old English Project - U of Toronto
Date: Sat, 2 Feb 1991 01:00:46 GMT

Thanks to the several people who mentioned my ommission of a file
in the dvi2tty post. Here it is:

David

=== dvi2tty.h === CUT HERE ===
#define Copyright "dvi2tty.c  Copyright (C) 1984, 1985, 1986 Svante Lindahl.\n\
Copyright (C) 1988 M.J.E. Mol"

#include <stdio.h>

#define TRUE        1
#define FALSE       0
#define nil         NULL

/*
 * ERROR CODES , don't start with 0
 */

#define illop    1              /* illegal op-code                  */
#define stkof    2              /* stack over-flow                  */
#define stkuf    3              /* stack under-flow                 */
#define stkrq    4              /* stack requirement                */
#define badid    5              /* id is not right                  */
#define bdsgn    6              /* signature is wrong               */
#define fwsgn    7              /* too few signatures               */
#define nopre    8              /* no pre-amble where expected      */
#define nobop    9              /* no bop-command where expected    */
```

```
#define nopp    10              /* no postpost where expected        */
#define bdpre   11              /* unexpected preamble occured       */
#define bdbop   12              /* unexpected bop-command occured    */
#define bdpst   13              /* unexpected post-command occured   */
#define bdpp    14              /* unexpected postpost               */
#define nopst   15              /* no post-amble where expected      */
#define illch   16              /* character code out of range       */
#define filop   17              /* cannot access file                */
#define filcr   18              /* cannot creat file                 */
#define pipcr   19              /* cannot creat pipe                 */

/*-----------------------------------------------------------------------*/

typedef char bool;

typedef struct prlistptr {      /* list of pages selected for output         */
    int       pag;                      /* number of pages                   */
    bool      all;                      /* pages in intervall selected       */
    struct prlistptr *prv;              /* previous item in list             */
    struct prlistptr *nxt;              /* next item in list                 */
    int       adr;                      /* nr of byte in file where bop is   */
} printlisttype;

/*-----------------------------------------------------------------------*/

extern bool   outputtofile;           /* output to file or stdout(dvi2tty.c)*/
extern bool   pageswitchon;           /* user-set pages to print(dvistuff.c)*/
extern bool   sequenceon;             /* not TeX pagenrs (dvistuff.c)       */
extern bool   scascii;                /* Scand. nat. chars (dvistuff.c)     */
extern bool   noffd;                  /* output ^L or formfeed (dvistuff.c) */

extern printlisttype *currentpage;    /* current page to print (dvi2tty.c)  */
extern printlisttype *firstpage;      /* first page selected (dvi2tty.c)    */
extern printlisttype *lastpage;       /* last page selected (dvi2tty.c)     */

extern int            ttywidth;       /* screen width (dvi2tty.c)           */
extern int            foo;            /* temporary 'register' (dvi2tty.c)   */
extern int            opcode;         /* dvi opcode (dvistuff.c)            */

extern FILE *DVIfile;                 /* dvi file (dvi2tty.c)               */
extern FILE *output;                  /* output file (dvi2tty.c)            */

/*-----------------------------------------------------------------------*/

/* dvi2tty.c */
void errorexit();

/* dvistuff.c */
void dvimain();
--
////////////////////////////////////////////////////////////////////
/  David Megginson                    david@doe.utoronto.ca         /
/  Centre for Medieval Studies        meggin@vm.epas.utoronto.ca    /
```

//

From 545037785936688546
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,de1b7d03c666b7b2,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,de1b7d03c666b7b2,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1991-03-03 12:47:16 PST
Path: gmdzi!unido!mcsun!uunet!spool.mu.edu!sdd.hp.com!cs.utexas.edu!sun-barr!
newstop!grapevine!panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st,comp.sys.atari.st
Subject: v05INF1: Introduction to comp.sources.atari.st
Message-ID: <967@grapevine.EBay.Sun.COM>
Date: 2 Mar 91 22:02:36 GMT
Expires: 30 Jun 91 22:02:36 GMT
Sender: news@grapevine.EBay.Sun.COM
Followup-To: comp.sources.atari.st
Lines: 155
Approved: koreth@panarthea.ebay.sun.com (Steven Grimm)
Xref: gmdzi comp.sources.atari.st:438 comp.sys.atari.st:31229
Posted: Sat Mar 2 16:02:36 1991

Submitted-by: koreth@panarthea.ebay.sun.com (Steven Grimm)
Posting-number: Volume 5, Info 1
Archive-name: intro

This is the first of two introductory articles about comp.sources.atari.st.
This one describes how to submit source to the newsgroup, where the
archive sites are, and how to contact them. The companion article lists
all previously-published sources.

I am always looking for suggestions on how to improve the usefulness
of the newsgroup, and can be contacted as listed below.

-- Steven Grimm
   koreth@panarthea.ebay.sun.com

Subject: Submitting source for publication

Items intended for posting or queries and problem notes should be sent to
atari-sources@panarthea.ebay.sun.com. If you are on a UUCP-only site, you
can send them to sun!ebay!panarthea!atari-sources. In Europe, send your
submissions to Jan-Hinrich Fessel, the European submoderator of the group,
at unido!atari-sources (or, if you're a masochist, atari-sources@unido.
informatik.uni-dortmund.de.)

If you want verification of arrival, so say in a cover note, or at the
beginning of your submission, if it is small. I try to verify that a

program works, and if I can't get it to work, I may hold up posting it for a couple of days. Please note that, except in rare cases, source without documentation will not be published. The backlog from receipt to posting varies from one to four weeks depending mostly on the set of submissions currently in my queue.

I prefer that you submit sources and binaries separately, as it's a pain to separate them, and such submissions are invariably placed on the back burner. Sources should be submitted in the same format as binaries -- uuencoded arc, zoo, or lzh files. Please use Pierre Dumas' "uue" program if possible, as it can correct some common transmission errors. Split long submissions up into reasonable (less than 50K) chunks. "uue" will do this for you, if you use "uue -700 file" to encode.

Please inform me of any special system requirements, and what development environment was used, so that I can properly test your submission and label it when I post it.

Subject: The structure of comp.sources.atari.st articles

Each posting in comp.sources.atari.st is called an "issue"; there are roughly 100 issues to a volume. The division is arbitrary and may vary. There are two types of articles in comp.sources.atari.st: sources and "information postings." They can be distinguished by the subject line:

  Subject: v05INF2: Index for Volume 1 and other info

This first word in the title identifies this as the third info posting of volume one. Similarly, the subject line shown below:

  Subject: v05i081: godsrc -- Universe simulator

identifies this as the 81st source article in Volume 5. Large sources are broken up into smaller pieces, and have subject lines that look like this:

  Subject: v05i088: decbustr -- VMS clone in ST-LOGO, Part03/08

Certain information about the system configuration required to use the program is given on the keywords line.

  Keywords: MWC, Megamax, 1meg, high

This means that the program compiles with Mark Williams or Megamax C, requires at least one meg of RAM and only runs in high resolution. Following is a list of keywords; new ones may be added as needed. They are mostly self-explanatory.

  Alcyon - Alcyon C (included with Atari developer's kit)
  MWC   - Mark Williams C

```
  Megamax - Megamax C
  Laser       - Laser C
  Lattice - Lattice C
  C       - Any C compiler
  OSS         - OSS Personal Pascal
  BASIC       - ST BASIC (this is not a joke!)
  uuencode    - program is uuencoded (uudecode required to unpack)
  arc         - program is archived (arc required to unpack)
  high        - high resolution
  medium  - medium resolution
  low         - low resolution
  1meg        - needs 1 meg of RAM
```

The References: line, if present, points to the previous part of a multi-part program, so that newsreaders (trn, nn, etc.) which can follow threads will recognize a large program as a single thread.

The first few lines of an article are auxiliary headers that look like this:

```
  Submitted-by: david@bdt.UUCP (David Beckmeyer)
  Posting-number: Volume 5, Issue 80
  Archive-name: micro-rtx
```

The "Submitted by" is the author of the program. If you have comments about the sources published in comp.sources.atari.st this is the person to contact. When possible, this address is in domain form, otherwise it is a UUCP bang path relative to some major (backbone) site.

The second line repeats the volume/issue information for the aide of notes sites and automatic archiving programs.

The Archive-name is the "official" name of this source in the archive. Large postings will have names that look like this:

```
Archive-name: desktop/part01
```

Please try to use this name when requesting that sources be mailed to you. Also, note that the "part number" given in the title, and the archive name given in the auxiliary header need not be identical.

Since most archive sites run UNIX, articles are given UNIX-style filenames rather than ST-style filenames. I do try to limit filenames to eight characters whenever possible, though.

Subject: Reporting and tracking bugs and patches to postings

Most of the time, bugs and updates will be posted to comp.sys.atari.st. Sometimes, "official" updates from the author will be posted submitted to comp.sources.atari.st so that they are archived with the sources.

Since there is no easy way to apply source code patches on the ST, changes

to a program often result in a whole new version being posted.

To report bugs, contact the person listed in the Submitted-to header.
Often there is a contact address in a README file, too. I do not maintain
the sources I moderate, so don't send your bug reports to me.

Subject: Listing of archive sites in no particular order

1. Site panarthea is running an archive server, which automatically answers
   mail requests. Send mail to archive-server@panarthea.ebay.sun.com with the
   word "help" in the message body for more information. Both the sources
   and binaries groups are archived on panarthea.
2. Unido is the European archive site. It's running an archive server
   similar to panarthea's; send "help" to unido!archive-server for more
   information. Unido archives both the sources and binaries groups.

If you know of additional sites which reliably archive comp.sources.atari.st
(or comp.binaries.atari.st, for that matter), please tell me so I can add
them to the list.

From 4202573135996663264
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,349c58077657d43e,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,349c58077657d43e,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1991-03-03 12:47:20 PST
Path: gmdzi!unido!mcsun!uunet!cs.utexas.edu!sun-barr!newstop!grapevine!
panarthea.ebay.sun.com
From: koreth@panarthea.ebay.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st,comp.sys.atari.st
Subject: v05INF2: Index of previously-published sources
Message-ID: <968@grapevine.EBay.Sun.COM>
Date: 2 Mar 91 22:02:42 GMT
Expires: 1 May 91 22:02:41 GMT
Sender: news@grapevine.EBay.Sun.COM
Followup-To: comp.sources.atari.st
Lines: 403
Approved: koreth@panarthea.ebay.sun.com
Xref: gmdzi comp.sources.atari.st:439 comp.sys.atari.st:31230

Submitted-by: koreth@panarthea.ebay.sun.com (Steven Grimm)
Posting-number: Volume 5, Info 2
Archive-name: Index

-- VOLUME 1 --

volume1/intro   Introduction to comp.sources.atari.st
volume1/basic-uud   Uudecode in BASIC
volume1/bootcopy   Automatic file copier for RAMdisk

```
volume1/joyread             Joystick-reading routines
volume1/cc/part01       C compiler driver (Alcyon)
volume1/cc/part02
volume1/cc/part03
volume1/cc/part04
volume1/mkmfor              Something for Absoft Fortran
volume1/make/part01         Program maintainer
volume1/make/part02
volume1/make/part03
volume1/make/part04
volume1/make/part05
volume1/make/part06
volume1/make/part07
volume1/blank           Fast disk eraser
volume1/mmdummy             Megamax dummy routines
volume1/fdum            Megamax floating-point dummy routines
volume1/idle            Screen saver
volume1/autotime        Time setter for auto folder
volume1/time            Time setter
volume1/autodisk        Copy files to RAMdisk
volume1/dvist32/part01      DVI previewer
volume1/dvist32/part02
volume1/dvist32/part03
volume1/dvist32/part04
volume1/dvist32/part05
volume1/dvist32/part06
volume1/dvist32/part07
volume1/dvist32/part08
volume1/dvist32/part09
volume1/cache           Disk cache
volume1/vdit            VDI interface with GDOS font support
volume1/flip/part01         Double-sided disk copier
volume1/flip/part02
volume1/rzsz/part01         UNIX end for zmdm, below
volume1/rzsz/part02
volume1/rzsz/part03
volume1/rzsz/part04
volume1/rzsz/part05
volume1/zmdm/part01         Zmodem file transfer program
volume1/zmdm/part02
volume1/zmdm/part03
volume1/zmdm/part04
volume1/zmdm/part05
volume1/zmdm/part06
volume1/zmdm/part07
volume1/zmdm/part08
volume1/ops5/part01         OPS5 production system in LISP
volume1/ops5/part02
volume1/ops5/part03
volume1/assist2/part01      Multifunction desk accessory
volume1/assist2/part02
volume1/errorchk        Verbose display of error codes
volume1/joinprnt        Echo console output to printer
```

```
volume1/color           Color-selection dialog box
volume1/rsctoc               Convert resource to C source
volume1/fastdump        Text-only screen dump
volume1/readme12        Online help system
volume1/newstart        Fancy printing
volume1/varlint
volume1/flip27/part01        Double-sided disk copier
volume1/flip27/part02
volume1/poplib               Shared libraries, with examples
volume1/uupc/part01          UUCP clone (pre-pre-release)
volume1/uupc/part02
volume1/uupc/part03
volume1/uupc/part04
volume1/gnufp           GNU floating point routines
volume1/scroll               Fast screen scroller
volume1/break           Stop programs with a keypress
volume1/newbreak        New version of above
volume1/timecode        Determine 68000 code speed
volume1/achlib/part01        Memory management and other routines for MadMac
volume1/achlib/part02
volume1/sttar           Unix-style "tar" utility
volume1/d110edit        D110 patch editor (MIDI)
volume1/raymovi2/part01      Raytrace movie generator
volume1/raymovi2/part02
volume1/gstlink              Faster linker for GST object files
volume1/mx2v230/part01       Multitasking kernel with utilities
volume1/mx2v230/part02
volume1/mx2v230/part03
volume1/mx2v230/part04
volume1/mx2v230/part05
volume1/mx2v230/part06
volume1/mx2v230/part07
volume1/mx2v230/part08
volume1/limiter              Limit uppermost hard disk device letter
volume1/d110edit        Patch editor for D110 synthesizer
volume1/mt32edit        Patch editor for MT32 synthesizer
volume1/starchrt/part01      Diffs to Unix starchart program for ST
volume1/starchrt/part02
volume1/passm/part01         Logic circuit assembler
volume1/passm/part02
volume1/leolife              Game of Life

-- VOLUME 2 --

volume2/sc_src/part01        Source to Sozobon C compiler
volume2/sc_src/part02
volume2/sc_src/part03
volume2/sc_src/part04
volume2/sc_src/part05
volume2/sc_src/part06
volume2/sc_src/part07
volume2/sc_src/part08
volume2/sc_src/part09
```

```
volume2/dlibs12/part01      Source for DLibs C libraries
volume2/dlibs12/part02
volume2/dlibs12/part03
volume2/shar            Pack and unpack shell archives
volume2/keyfix              Fix Gulam for non-American keyboards
volume2/sed/part01      GNU sed (stream editor)
volume2/sed/part02
volume2/sort/part01         UNIX-style sort utility
volume2/sort/part02
volume2/uw/part01       Unix Windows windowing terminal software
volume2/uw/part02
volume2/uw/part03
volume2/uw/part04
volume2/uw/part05
volume2/uuduue              Dumas uuencode/decode programs
volume2/mwstring        dpANS string.h for MWC
volume2/mailtruk        GFA BASIC uucp software
volume2/format83        Make Intel ASCII Hex files
volume2/mailtruk.11         Version 1.1 of GFA-BASIC uucp software
volume2/1stjet              1st-Word configuration file for DeskJet
volume2/1stjethv        1st-Word config file to use helvetica font on DeskJet
volume2/cpboot              Copy executable boot sectors between floppy formats
volume2/du              List disk usage in directory trees
volume2/fplib10             Floating point routines for Sozobon C
volume2/prtset              Set printer parameters
volume2/hexify              Hexidecimal encode/decode for mailing binaries
volume2/nutshell        Use a shell from within a GEM prog
volume2/texchang        Change files for BibTeX, Tangle, and Weave
volume2/deskset             Re-read DESKTOP.INF and set up system accordingly
volume2/which           Find programs using PATH env. variable
volume2/atime89             Set GEMDOS time from keyboard clock
volume2/fplib20/part01      New floating-point library for Sozobon C
volume2/fplib20/part02
volume2/decky           Connect DEC keyboard to ST
volume2/djreset             Reset DeskJet Plus
volume2/modedit             Hardwire an editor into German Modula-2
volume2/overscan        Software driver for increased screen size
volume2/gnutar/part01       GNU's version of UNIX "tar"
volume2/gnutar/part02
volume2/gnutar/part03
volume2/gnutar/part04
volume2/gnutar/part05
volume2/gnutar/part06
volume2/gnutar/part07
volume2/gnutar/part08
volume2/coloremu        Color emulator for mono screens
volume2/gccgem/part01       GEM libraries for GCC
volume2/gccgem/part02
volume2/gccgem/part03
volume2/gccgem/part04
volume2/gccgem/part05
volume2/gccgem/part06
volume2/postmort        Analyze post-bomb information
```

```
volume2/crunch              Executable file compressor
volume2/environ             Set environment variables from AUTO folder
volume2/viruskil       Virus killer
volume2/norad          Earth satellite tracking program
volume2/diff           File difference utility
volume2/emacs.diff/part01   Diffs to GNU-EMACS 18.52 for Atari
volume2/emacs.diff/part02
volume2/protect             Virus killer/protector
volume2/ff31           File finder version 3.1
volume2/gcclib2/part01      Patches to GCC library
volume2/gcclib2/part02
volume2/gcclib2/part03
volume2/gcclib2/part04
volume2/gcclib2/part05
volume2/gcclib2/part06
volume2/gcclib2/part07
volume2/gcclib2-patch1      Patch for GCC library
volume2/mandel              Fast Mandelbrot set generator
volume2/gfadiff             File difference generator
volume2/nolock              Disable Caps-Lock key
volume2/builds09       Build Motorola S-records from .ttp files
volume2/colorca             Cellular automata in color
volume2/eternal3       Reset-proof ramdisk
volume2/degadesk       Load DEGAS .P?3 files onto desktop
volume2/truchet             Display random Truchet tilings on mono screens
volume2/yaff           Yet Another File Finder
volume2/strings             Find text strings in files
volume2/curses/part01       Screen updating package
volume2/curses/part02
volume2/sozobon1.2/part01   Update to Sozobon C compiler
volume2/sozobon1.2/part02
volume2/sozobon1.2/part03
volume2/sozobon1.2/part04
volume2/sozobon1.2/part05
volume2/sozobon1.2/part06
volume2/sozobon1.2/part07
volume2/sozobon1.2/part08
volume2/sozobon1.2/part09

-- VOLUME 3 --

volume3/fract          Monochrome Mandelbrot generator
volume3/uurename       Rename binary postings to work with uud
volume3/uustuff             More uud-ing tools
volume3/uustuff2       Addendum to previous posting
volume3/xxed/part01         Hex/ASCII file editor
volume3/xxed/part02
volume3/xxed/part03
volume3/xxed/patch01        Messed-up files from XXED
volume3/nroff/part01        Partial clone of UNIX "nroff" formatter
volume3/nroff/part02
volume3/nroff/part03
volume3/kbext410       Keyboard remapper
```

```
volume3/cal             UNIXoid calendar program
volume3/dhrystone       Dhrystone benchmark program
volume3/quix            Game: fill up the screen
volume3/3demo/part01        3-D graphics demo
volume3/3demo/part02
volume3/xxedpat             Messed-up files from XXED
volume3/tosvers             Tell TOS version number
volume3/lharc/part01        LHARC archiver
volume3/lharc/part02
volume3/sound           Sound generation docs, plus sample sounds
volume3/necp6           Screendump programs for NEC P6 printer
volume3/file/part01         Determine file types
volume3/file/part02
volume3/barcodes        Metafont file for printing barcodes
volume3/mx2net20        Share disk drives across MIDI/RS232
volume3/splitit             Split up and reassemble big files
volume3/bm              Fast fixed-string searcher
volume3/abzmon              Machine-language monitor
volume3/metaconv        Convert EZDraw/UniTerm metafiles to GEM format
volume3/netsrc              The REAL sources to mx2net20, MIDI/RS232 network
volume3/szadb/part01        Debugger for Sozobon C
volume3/szadb/part02
volume3/dstools/part01      Data structure toolkit
volume3/dstools/part02
volume3/sam             Disk analyzer
volume3/sre             Run programs when gulam is started up
volume3/lharc113/part01     LHArc archiver, version 1.13
volume3/lharc113/part02
volume3/lharc113/part03
volume3/jove/part01         Emacs variant, v. 4.9
volume3/jove/part02
volume3/jove/part03
volume3/jove/part04
volume3/jove/part05
volume3/jove/part06
volume3/jove/part07
volume3/jove/part08
volume3/jove/part09
volume3/jove/part10
volume3/jove/part11
volume3/jove/part12
volume3/xxed11/part01       XXED editor, version 1.1
volume3/xxed11/part02
volume3/xbra            XBRA trap-catching system
volume3/wd              Uudecoding program (takes multiple file args)
volume3/gfa-uucode      Uuencode/decode program in GFA BASIC
volume3/sozoinit        Startup module for Sozobon C, for accs and prgs
volume3/fgrep           GNU fgrep
volume3/belltest        Test out NewBell bell sounds
volume3/chaos           Draw random Sierpinski Triangles
volume3/bifurc              Draw bifurcation diagrams
volume3/vbi             Vertical blank interrupt handler (w/example)
volume3/sunclock        Watch the sun go around a map of the Earth
```

```
volume3/norad/part01        Satellite tracker, version 2
volume3/norad/part02
volume3/st_conc             Concentration game
volume3/mkrsc/part01        Resource construction program
volume3/mkrsc/part02
volume3/mkrsc/part03
volume3/mines           Game of Mines for mono systems
volume3/showspec        Show Spectrum .SPU pictures
volume3/switcher        Pick random NeoDesk/DegaDesk pics, NewBell sounds
volume3/maze            Walk through a maze in 3-D
volume3/nroff1.10/part01        Nroff clone, version 1.10
volume3/nroff1.10/part02
volume3/nroff1.10/part03
volume3/pi1tomon        Display .PI1 pictures on a mono monitor
volume3/gulwrap             Make Gulam work under Gemini
volume3/dublvis             Format DS disks that work in SS drives
volume3/grep15/part01       GNU grep and fgrep, version 1.5
volume3/grep15/part02
volume3/grep15/part03
volume3/xmore           Configurable "more" for various file types
volume3/mutate              Disk sector editor
volume3/xxxx            Insert numbers into files
volume3/sozo1.2pat      Patches for Sozobon C 1.2
volume3/unx2st              Convert ST <-> UNIX end-of-lines
volume3/mtprog1             Multitasking in Programming tutorial, posting 1
volume3/nroff110pat         Patch for Nroff 1.10
volume3/lharc1.02/part01    Lharc source for ST and UNIX
volume3/lharc1.02/part02
volume3/lharc1.02/part03
volume3/cat2asc             Convert STDCAT43 catalogs to pretty ASCII form
volume3/nroff110pat2        Patch 2 for nroff v. 1.10
volume3/nrchbar             Generate change bars in n/troff documents
volume3/stturing        Two-dimensional Turing machine
volume3/hdwboot             Wait for hard disk to come online

-- VOLUME 4 --

volume4/mint06s/part01      UNIXlike multitasking system
volume4/mint06s/part02
volume4/mint06s/part03
volume4/mint06s/part04
volume4/mint06s/part05
volume4/mint06s/part06
volume4/mint06s/part07
volume4/mint06s/part08
volume4/mint06s/part09
volume4/mint06s/part10
volume4/mint06s/part11
volume4/mgrdiffs        Diffs to MGR window manager, to run under MiNT
volume4/bgacc      Desk acc. to run background jobs under MiNT
volume4/fcat       Fast "cat" for Gulam
volume4/cdecl/part01        Generate C declarations
volume4/cdecl/part02
```

```
volume4/fcat11          Faster "cat" for Gulam, v 1.1
volume4/popen      popen(3) call for plain TOS
volume4/eplay20         STe sound sample player
volume4/etext      Filter ASCII files into PageStream format
volume4/gulmake         Files for compiling gcc under Gulam
volume4/gnum4/part01        GNU "m4" macro processor, v 0.50
volume4/gnum4/part02
volume4/gnum4/part03
volume4/gnum4/part04
volume4/gnum4/part05
volume4/mkrsc/part01        Resource construction program
volume4/mkrsc/part02
volume4/mkrsc/part03
volume4/cdecl-p1        Generate C declarations, patch 1
volume4/mgif       GIF viewer for monochrome STs
volume4/gawk211/part01      GNU awk, version 2.11
volume4/gawk211/part02
volume4/gawk211/part03
volume4/gawk211/part04
volume4/gawk211/part05
volume4/gawk211/part06
volume4/gawk211/part07
volume4/motasm          Motorola assemblers for 68xx processors
volume4/xlharc12/part01     Another version of lharc (based on UNIX lharc 0.03)
volume4/xlharc12/part02
volume4/sozosun         Diffs to Sozobon C for cross-compiling on Suns
volume4/zlib       Library functions for reading .Z files
volume4/killdrv         Add/remove drives from TOS's drive list
volume4/szadb12/part01      Sozobon adb debugger, v 1.2
volume4/szadb12/part02
volume4/szadb12/part03
volume4/pml_sqrt        Fast square root function for gcc
volume4/gemfstfx        Fixes for vdifast from Gemfast 1.3
volume4/xstrip          eXtended Strip, for removing symbol tables
volume4/lynxlib/part01      Library of miscellaneous routines
volume4/lynxlib/part02
volume4/lynxlib/part03
volume4/lynxlib/part04
volume4/mclock          Corner clock for mono systems
volume4/mgif11          GIF viewer for mono systems, v 1.1
volume4/nlight          Show I/O to any disk by flashing floppy light
volume4/less97/part01       file pager for MiNT
volume4/less97/part02
volume4/less97/part03
volume4/less97/part04
volume4/pathalias/part01        Build UUCP paths file from maps
volume4/pathalias/part02
volume4/pathalias/part03
volume4/dvi2tty         Plain text dvi previewer
volume4/mercury/part01      Mercury UUCP: mail/news package
volume4/mercury/part02
volume4/mercury/part03
volume4/mercury/part04
```

volume4/mercury/part05
volume4/mercury/part06
volume4/nroff110p4/part01   Nroff clone, version 1.10 patchlevel 4
volume4/nroff110p4/part02
volume4/nroff110p4/part03
volume4/nroff110p4/part04
volume4/man   Read online manual pages
volume4/manpg14/part01   Pager for reading online manual pages
volume4/manpg14/part02
volume4/manpg14/part03
volume4/dvithead   dvi2tty.h (missing file from dvi2tty)
volume4/diff115/part01   GNU diff, version 1.15
volume4/diff115/part02
volume4/diff115/part03
volume4/diff115/part04
volume4/rcs5ap1/part01   Revision Control System, version 5
volume4/rcs5ap1/part02
volume4/rcs5ap1/part03
volume4/rcs5ap1/part04
volume4/rcs5ap1/part05
volume4/rcs5ap1/part06
volume4/rcs5ap1/part07
volume4/rcs5ap1/part08

From -2437976795598693624
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,49acfd40ab7d4d08,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,49acfd40ab7d4d08,start
X-Google-Attributes: gid100493,public
X-Google-Thread: 109a67,49acfd40ab7d4d08,start
X-Google-Attributes: gid109a67,public
X-Google-ArrivalTime: 1991-03-25 08:14:23 PST
Xref: utzoo comp.sys.atari.st:36527 comp.sys.atari.st.tech:1776
comp.sources.atari.st:463
Newsgroups: comp.sys.atari.st,comp.sys.atari.st.tech,comp.sources.atari.st
Path: utzoo!utdoe!david
From: david@doe.utoronto.ca (David Megginson)
Subject: PATCH for tr(1v) to fix bug (please apply)
Message-ID: <1991Mar25.161423.24739@doe.utoronto.ca>
Followup-To: comp.sys.atari.st
Summary: tr prints out the last buffer twice!
Sender: david@doe.utoronto.ca (David Megginson)
Organization: Dictionary of Old English Project - U of Toronto
Date: Mon, 25 Mar 1991 16:14:23 GMT

I have found a bug in my tr(1v) clone for MiNT--under some circumstances,
the program will write out the last buffer twice. I am posting this patch
to comp.sys.atari.st and comp.sources.atari.st, and I will post the full
fixed sources and binaries in a week or so (they should take another week
to appear). Until then, could anyone using my tr or wc please send in any
other bugs, comments or requests, so that I can (maybe) incorporate them?

Thank you.

David

Apply this patch to tr.c:

======CUT HERE======

```
16a17
> /*      -DDEBUG            Trace buffer handling  */
20c21
< /* $Id: tr.c,v 1.2 1991/03/05 09:01:38 david Exp $ */
---
> /* $Id: tr.c,v 1.3 1991/03/25 07:51:28 david Exp $ */
23a25,27
>  * Revision 1.3  1991/03/25  07:51:28  david
>  * Fixed error in do_translate() which output last buffer twice.
>  *
32c36
< static char * version = "$Id: tr.c,v 1.2 1991/03/05 09:01:38 david Exp $";
---
> static char * version = "$Id: tr.c,v 1.3 1991/03/25 07:51:28 david Exp $";
342c346
<         if( total_read > 0 )
---
>         if( total_read > 0 ) {
344c348,352
<         write(1,buffer2,total_write);
---
>             write(1,buffer2,total_write);
>         }
> #ifdef DEBUG
> fprintf(stderr,"Total read: %ld, Total write: %ld\n",total_read,total_write);
> #endif
--
```


//
/ David Megginson   david@doe.utoronto.ca   /
/ Centre for Medieval Studies   meggin@vm.epas.utoronto.ca   /
//

From 3561668108216888555
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,1c160837d3923da3,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: f9437,1c160837d3923da3,start
X-Google-Attributes: gidf9437,public
X-Google-Thread: 100493,1c160837d3923da3,start
X-Google-Attributes: gid100493,public
X-Google-Thread: 156df7,1c160837d3923da3,start
X-Google-Attributes: gid156df7,public
X-Google-ArrivalTime: 1991-05-09 02:53:28 PST
Path: gmdzi!unido!math.fu-berlin.de!fub!ox.com!caen!sdd.hp.com!cs.utexas.edu!sun-

barr!newstop!sun!exodus!twitterpater.Eng.Sun.COM!koreth
From: koreth@twitterpater.Eng.Sun.COM (Steven Grimm)
Newsgroups:
comp.binaries.atari.st,comp.sources.atari.st,comp.sys.atari.st,news.groups
Subject: Administrivia: "panarthea" is now "twitterpater" (comp.*.atari.st)
Message-ID: <13032@exodus.Eng.Sun.COM>
Date: 9 May 91 03:48:28 GMT
Sender: news@exodus.Eng.Sun.COM
Reply-To: koreth@twitterpater.Eng.Sun.COM (Steven Grimm)
Followup-To: comp.binaries.atari.st
Organization: Sun Microsystems, Inc. Mt. View, Ca.
Lines: 24
Approved: koreth@twitterpater.Eng.Sun.COM (Steven Grimm)
Xref: gmdzi comp.binaries.atari.st:1461 comp.sources.atari.st:445
comp.sys.atari.st:82888 news.groups:137936

I've moved locations at Sun, and my machine came with me. I've moved all my
moderation gear over to the new location, so comp.sources.atari.st and
comp.binaries.atari.st submissions should be sent here instead of to the old
addresses. The new addresses are:

atari-sources@twitterpater.eng.sun.com for comp.sources.atari.st submissions
atari-binaries@twitterpater.eng.sun.com for comp.binaries.atari.st submissions

The panarthea archive server has also been moved to twitterpater. Its
address, as you can probably guess, is now

archive-server@twitterpater.eng.sun.com

The panarthea addresses for all of the above will continue to work for a while,
but they'll forward everything to twitterpater. If you maintain a list of
archive sites, please note the change.

Please let me know if there are any problems. I've tested everything and think
I have things completely moved over, but I might have missed something.

Steven Grimm koreth@eng.sun.com   Moderator, comp.{sources,binaries}.atari.st
"We must be brave, and not let them know how frightened we really are."
                -- OPEN LOOK Graphical User Interface Functional Specification

From -8919358215539280015
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,eb0ccac4fedd65bf,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1992-03-06 14:47:16 PST
Path: sparky!uunet!think.com!ames!sun-barr!male.EBay.Sun.COM!exodus.Eng.Sun.COM!
exodus.Eng.Sun.COM!news
From: koreth@twitterpater.eng.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st
Subject: v05i083: emacs.rc -- Micro-EMACS startup file for C source editing
Keywords: emacs

Message-ID: <sources.5.83@twitterpater.eng.sun.com>
Date: 6 Mar 92 22:47:16 GMT
Reply-To: elrond@frmug.fr.mugnet.org (Bertrand Petit)
Organization: Sun Microsystems Inc., Mountain View, CA
Lines: 115
Approved: koreth@twitterpater.eng.sun.com
NNTP-Posting-Host: twitterpater

Submitted-by: elrond@frmug.fr.mugnet.org (Bertrand Petit)
Posting-number: Volume 5, Issue 83
Archive-name: emacs.rc

[Docs in the comments immediately below. -sg]

------------------------start of emacs.rc-----------------------------------
```
;#####################################################################################
;
; This emacs.rc file can be used by Atari ST owners who whan to use the
; facilities of micro emacs in C source code editing.
;
; I have written this file for my own use but it's so usefull I have decided
; to bring this to the public eyes. You can distribute this file since he
; remain unaltered, no charge can be assked and my name and e-mail adress
; musn't be deleted nor modified. I you wan to distribute a modified version
; give me the credit for the original version and send me a copy at the
; e-mail adress found bellow.
;
; A file named "ERROR" must be loaded, the screen will be splited in two
; windows. Just press M-E on an error or warning line preduced by TCC to load
; the aproriate file at the line of the error.
;
; Some work can be done on this macro:
;    -we can check if the cursor really on an error line
;    -the find-file command can fail but such errors are not handled.
;
;
; --
; Bertrand Petit alias >Elrond the half-Elf<
; e-mail: elrond@frmug.gna.tfd.com
;         elrond@frmug.mugnet.fr.org
;#####################################################################################


;##########################
;# set screen behavior #
;##########################
set $sscroll TRUE
set $hscroll TRUE
set $hjump 1

add-global-mode "blue"

;#######################################
```

```
;# define special keys and behavior #
;#####################################
set $readhook execute-macro-20
bind-to-key execute-macro-22 M-E

;#########################################################################################
;# This macro set modes and/or window according to the file type #
;#########################################################################################
20   store-macro

     ;#### Prepare windows for TCC output
     !if &or &seq $cfname "ERROR" &seq $cfname "error"
          split-current-window
          next-window
          "empty" select-buffer
          previous-window
     !endif

     ;#### Get the file extention
     set %rctmp &sin $cfname "."
     !if &equ %rctmp 0
          !return
     !endif
     set %rctmp &mid $cfname &add %rctmp 1 5

     ;#### Is it a C source code?
     !if &or &seq %rctmp "c" &seq %rctmp "h"
          add-mode "cmode"
          set $hardtab 4
          set $tpause 10
     !endif
!endm

;#########################################################################################
;# This macro, acts when M-E is pressed, load a C source code and go to the #
;# line according to what is found on the current line. The buffer must     #
;# contain a TCC output.                                                    #
;#########################################################################################
22   store-macro
     ;#### Is it an error file?
     !if &seq $cbufname "error"
          write-message "[Loading source file]"
          set %oldcol $curcol
          ;#### Searching for the file name
          beginning-of-line
          next-word
          set %deb &add $curcol 1
          end-of-word
          end-of-word
          set %file &mid $line %deb &add &sub $curcol %deb 1
          ;#### Searching for the line number
          next-word
          set %deb &add $curcol 1
```

```
          end-of-word
          set %line &mid $line %deb &add &sub $curcol %deb 1
          set $curcol %oldcol
          ;#### Loading and displaying of the file
          previous-window
          find-file %file
          goto-line %line
     !else
          write-message "[This is not an error buffer]"
     !endif
!endm
--------------------end of emacs.rc----------------------------------------

--
|    Bertrand Petit          |         Signature is under construction         |
|    alias                   |                                                 |
|    >Elrond le demi-Elfe<   |         Be sure to wear your helmet             |


From 3871907297364953434
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,a9bec690ff17c206,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 100493,a9bec690ff17c206,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1992-06-29 04:07:10 PST
Xref: sparky comp.sources.atari.st:54 comp.sys.atari.st:10668
Path: sparky!uunet!sun-barr!news2me.ebay.sun.com!exodus.Eng.Sun.COM!
appserv.Eng.Sun.COM!appserv.Eng.Sun.COM!news
From: koreth@twitterpater.eng.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st,comp.sys.atari.st
Subject: v06INF2: Index of previously-published sources
Date: 30 Jun 1992 01:01:35 GMT
Organization: Sun Microsystems Inc., Mountain View, CA, USA
Lines: 510
Approved: koreth@twitterpater.eng.sun.com
Expires: 60days
Message-ID: <l4vcjfINNt42@appserv.Eng.Sun.COM>
NNTP-Posting-Host: twitterpater

Submitted-by: koreth@twitterpater.eng.sun.com (Steven Grimm)
Posting-number: Volume 6, Info 2
Archive-name: Index

-- Volume 1 --

volume1/intro          Introduction to comp.sources.atari.st
volume1/basic-uud      Uudecode in BASIC
volume1/bootcopy       Automatic file copier for RAMdisk
volume1/joyread             Joystick-reading routines
volume1/cc/part01      C compiler driver (Alcyon)
volume1/cc/part02
volume1/cc/part03
```

```
volume1/cc/part04
volume1/mkmfor              Something for Absoft Fortran
volume1/make/part01         Program maintainer
volume1/make/part02
volume1/make/part03
volume1/make/part04
volume1/make/part05
volume1/make/part06
volume1/make/part07
volume1/blank           Fast disk eraser
volume1/mmdummy             Megamax dummy routines
volume1/fdum            Megamax floating-point dummy routines
volume1/idle            Screen saver
volume1/autotime        Time setter for auto folder
volume1/time            Time setter
volume1/autodisk        Copy files to RAMdisk
volume1/dvist32/part01      DVI previewer
volume1/dvist32/part02
volume1/dvist32/part03
volume1/dvist32/part04
volume1/dvist32/part05
volume1/dvist32/part06
volume1/dvist32/part07
volume1/dvist32/part08
volume1/dvist32/part09
volume1/cache           Disk cache
volume1/vdit            VDI interface with GDOS font support
volume1/flip/part01         Double-sided disk copier
volume1/flip/part02
volume1/rzsz/part01         UNIX end for zmdm, below
volume1/rzsz/part02
volume1/rzsz/part03
volume1/rzsz/part04
volume1/rzsz/part05
volume1/zmdm/part01         Zmodem file transfer program
volume1/zmdm/part02
volume1/zmdm/part03
volume1/zmdm/part04
volume1/zmdm/part05
volume1/zmdm/part06
volume1/zmdm/part07
volume1/zmdm/part08
volume1/ops5/part01         OPS5 production system in LISP
volume1/ops5/part02
volume1/ops5/part03
volume1/assist2/part01      Multifunction desk accessory
volume1/assist2/part02
volume1/errorchk        Verbose display of error codes
volume1/joinprnt        Echo console output to printer
volume1/color           Color-selection dialog box
volume1/rsctoc              Convert resource to C source
volume1/fastdump        Text-only screen dump
volume1/readme12        Online help system
```

```
volume1/newstart       Fancy printing
volume1/varlint
volume1/flip27/part01       Double-sided disk copier
volume1/flip27/part02
volume1/poplib              Shared libraries, with examples
volume1/uupc/part01         UUCP clone (pre-pre-release)
volume1/uupc/part02
volume1/uupc/part03
volume1/uupc/part04
volume1/gnufp          GNU floating point routines
volume1/scroll              Fast screen scroller
volume1/break          Stop programs with a keypress
volume1/newbreak       New version of above
volume1/timecode       Determine 68000 code speed
volume1/achlib/part01       Memory management and other routines for MadMac
volume1/achlib/part02
volume1/sttar          Unix-style "tar" utility
volume1/d110edit       D110 patch editor (MIDI)
volume1/raymovi2/part01     Raytrace movie generator
volume1/raymovi2/part02
volume1/gstlink             Faster linker for GST object files
volume1/mx2v230/part01      Multitasking kernel with utilities
volume1/mx2v230/part02
volume1/mx2v230/part03
volume1/mx2v230/part04
volume1/mx2v230/part05
volume1/mx2v230/part06
volume1/mx2v230/part07
volume1/mx2v230/part08
volume1/limiter             Limit uppermost hard disk device letter
volume1/d110edit       Patch editor for D110 synthesizer
volume1/mt32edit       Patch editor for MT32 synthesizer
volume1/starchrt/part01     Diffs to Unix starchart program for ST
volume1/starchrt/part02
volume1/passm/part01        Logic circuit assembler
volume1/passm/part02
volume1/leolife             Game of Life

-- Volume 2 --

volume2/sc_src/part01       Source to Sozobon C compiler
volume2/sc_src/part02
volume2/sc_src/part03
volume2/sc_src/part04
volume2/sc_src/part05
volume2/sc_src/part06
volume2/sc_src/part07
volume2/sc_src/part08
volume2/sc_src/part09
volume2/dlibs12/part01      Source for DLibs C libraries
volume2/dlibs12/part02
volume2/dlibs12/part03
volume2/shar           Pack and unpack shell archives
```

```
volume2/keyfix             Fix Gulam for non-American keyboards
volume2/sed/part01     GNU sed (stream editor)
volume2/sed/part02
volume2/sort/part01        UNIX-style sort utility
volume2/sort/part02
volume2/uw/part01      Unix Windows windowing terminal software
volume2/uw/part02
volume2/uw/part03
volume2/uw/part04
volume2/uw/part05
volume2/uuduue             Dumas uuencode/decode programs
volume2/mwstring       dpANS string.h for MWC
volume2/mailtruk       GFA BASIC uucp software
volume2/format83       Make Intel ASCII Hex files
volume2/mailtruk.11        Version 1.1 of GFA-BASIC uucp software
volume2/1stjet             1st-Word configuration file for DeskJet
volume2/1stjethv       1st-Word config file to use helvetica font on DeskJet
volume2/cpboot             Copy executable boot sectors between floppy formats
volume2/du             List disk usage in directory trees
volume2/fplib10            Floating point routines for Sozobon C
volume2/prtset             Set printer parameters
volume2/hexify             Hexidecimal encode/decode for mailing binaries
volume2/nutshell       Use a shell from within a GEM prog
volume2/texchang       Change files for BibTeX, Tangle, and Weave
volume2/deskset            Re-read DESKTOP.INF and set up system accordingly
volume2/which          Find programs using PATH env. variable
volume2/atime89            Set GEMDOS time from keyboard clock
volume2/fplib20/part01     New floating-point library for Sozobon C
volume2/fplib20/part02
volume2/decky          Connect DEC keyboard to ST
volume2/djreset            Reset DeskJet Plus
volume2/modedit            Hardwire an editor into German Modula-2
volume2/overscan       Software driver for increased screen size
volume2/gnutar/part01      GNU's version of UNIX "tar"
volume2/gnutar/part02
volume2/gnutar/part03
volume2/gnutar/part04
volume2/gnutar/part05
volume2/gnutar/part06
volume2/gnutar/part07
volume2/gnutar/part08
volume2/coloremu       Color emulator for mono screens
volume2/gccgem/part01      GEM libraries for GCC
volume2/gccgem/part02
volume2/gccgem/part03
volume2/gccgem/part04
volume2/gccgem/part05
volume2/gccgem/part06
volume2/postmort       Analyze post-bomb information
volume2/crunch             Executable file compressor
volume2/environ            Set environment variables from AUTO folder
volume2/viruskil       Virus killer
volume2/norad          Earth satellite tracking program
```

```
volume2/diff            File difference utility
volume2/emacs.diff/part01   Diffs to GNU-EMACS 18.52 for Atari
volume2/emacs.diff/part02
volume2/protect             Virus killer/protector
volume2/ff31            File finder version 3.1
volume2/gcclib2/part01      Patches to GCC library
volume2/gcclib2/part02
volume2/gcclib2/part03
volume2/gcclib2/part04
volume2/gcclib2/part05
volume2/gcclib2/part06
volume2/gcclib2/part07
volume2/gcclib2-patch1      Patch for GCC library
volume2/mandel              Fast Mandelbrot set generator
volume2/gfadiff             File difference generator
volume2/nolock              Disable Caps-Lock key
volume2/builds09        Build Motorola S-records from .ttp files
volume2/colorca             Cellular automata in color
volume2/eternal3        Reset-proof ramdisk
volume2/degadesk        Load DEGAS .P?3 files onto desktop
volume2/truchet             Display random Truchet tilings on mono screens
volume2/yaff            Yet Another File Finder
volume2/strings             Find text strings in files
volume2/curses/part01       Screen updating package
volume2/curses/part02
volume2/sozobon1.2/part01   Update to Sozobon C compiler
volume2/sozobon1.2/part02
volume2/sozobon1.2/part03
volume2/sozobon1.2/part04
volume2/sozobon1.2/part05
volume2/sozobon1.2/part06
volume2/sozobon1.2/part07
volume2/sozobon1.2/part08
volume2/sozobon1.2/part09

-- Volume 3 --

volume3/fract           Monochrome Mandelbrot generator
volume3/uurename        Rename binary postings to work with uud
volume3/uustuff             More uud-ing tools
volume3/uustuff2        Addendum to previous posting
volume3/xxed/part01         Hex/ASCII file editor
volume3/xxed/part02
volume3/xxed/part03
volume3/xxed/patch01        Messed-up files from XXED
volume3/nroff/part01        Partial clone of UNIX "nroff" formatter
volume3/nroff/part02
volume3/nroff/part03
volume3/kbext410        Keyboard remapper
volume3/cal             UNIXoid calendar program
volume3/dhrystone       Dhrystone benchmark program
volume3/quix            Game: fill up the screen
volume3/3demo/part01        3-D graphics demo
```

```
volume3/3demo/part02
volume3/xxedpat             Messed-up files from XXED
volume3/tosvers             Tell TOS version number
volume3/lharc/part01        LHARC archiver
volume3/lharc/part02
volume3/sound           Sound generation docs, plus sample sounds
volume3/necp6           Screendump programs for NEC P6 printer
volume3/file/part01         Determine file types
volume3/file/part02
volume3/barcodes        Metafont file for printing barcodes
volume3/mx2net20        Share disk drives across MIDI/RS232
volume3/splitit             Split up and reassemble big files
volume3/bm              Fast fixed-string searcher
volume3/abzmon              Machine-language monitor
volume3/metaconv        Convert EZDraw/UniTerm metafiles to GEM format
volume3/netsrc              The REAL sources to mx2net20, MIDI/RS232 network
volume3/szadb/part01        Debugger for Sozobon C
volume3/szadb/part02
volume3/dstools/part01      Data structure toolkit
volume3/dstools/part02
volume3/sam             Disk analyzer
volume3/sre             Run programs when gulam is started up
volume3/lharc113/part01     LHArc archiver, version 1.13
volume3/lharc113/part02
volume3/lharc113/part03
volume3/jove/part01         Emacs variant, v. 4.9
volume3/jove/part02
volume3/jove/part03
volume3/jove/part04
volume3/jove/part05
volume3/jove/part06
volume3/jove/part07
volume3/jove/part08
volume3/jove/part09
volume3/jove/part10
volume3/jove/part11
volume3/jove/part12
volume3/xxed11/part01       XXED editor, version 1.1
volume3/xxed11/part02
volume3/xbra            XBRA trap-catching system
volume3/wd              Uudecoding program (takes multiple file args)
volume3/gfa-uucode      Uuencode/decode program in GFA BASIC
volume3/sozoinit        Startup module for Sozobon C, for accs and prgs
volume3/fgrep           GNU fgrep
volume3/belltest        Test out NewBell bell sounds
volume3/chaos           Draw random Sierpinski Triangles
volume3/bifurc              Draw bifurcation diagrams
volume3/vbi             Vertical blank interrupt handler (w/example)
volume3/sunclock        Watch the sun go around a map of the Earth
volume3/norad/part01        Satellite tracker, version 2
volume3/norad/part02
volume3/st_conc             Concentration game
volume3/mkrsc/part01        Resource construction program
```

```
volume3/mkrsc/part02
volume3/mkrsc/part03
volume3/mines           Game of Mines for mono systems
volume3/showspec        Show Spectrum .SPU pictures
volume3/switcher        Pick random NeoDesk/DegaDesk pics, NewBell sounds
volume3/maze            Walk through a maze in 3-D
volume3/nroff1.10/part01        Nroff clone, version 1.10
volume3/nroff1.10/part02
volume3/nroff1.10/part03
volume3/pi1tomon        Display .PI1 pictures on a mono monitor
volume3/gulwrap             Make Gulam work under Gemini
volume3/dublvis             Format DS disks that work in SS drives
volume3/grep15/part01       GNU grep and fgrep, version 1.5
volume3/grep15/part02
volume3/grep15/part03
volume3/xmore           Configurable "more" for various file types
volume3/mutate              Disk sector editor
volume3/xxxx            Insert numbers into files
volume3/sozo1.2pat      Patches for Sozobon C 1.2
volume3/unx2st              Convert ST <-> UNIX end-of-lines
volume3/mtprog1             Multitasking in Programming tutorial, posting 1
volume3/nroff110pat         Patch for Nroff 1.10
volume3/lharc1.02/part01    Lharc source for ST and UNIX
volume3/lharc1.02/part02
volume3/lharc1.02/part03
volume3/cat2asc             Convert STDCAT43 catalogs to pretty ASCII form
volume3/nroff110pat2        Patch 2 for nroff v. 1.10
volume3/nrchbar             Generate change bars in n/troff documents
volume3/stturing        Two-dimensional Turing machine
volume3/hdwboot             Wait for hard disk to come online

-- Volume 4 --

volume4/mint06s/part01      UNIXlike multitasking system
volume4/mint06s/part02
volume4/mint06s/part03
volume4/mint06s/part04
volume4/mint06s/part05
volume4/mint06s/part06
volume4/mint06s/part07
volume4/mint06s/part08
volume4/mint06s/part09
volume4/mint06s/part10
volume4/mint06s/part11
volume4/mgrdiffs        Diffs to MGR window manager, to run under MiNT
volume4/bgacc      Desk acc. to run background jobs under MiNT
volume4/fcat       Fast "cat" for Gulam
volume4/cdecl/part01        Generate C declarations
volume4/cdecl/part02
volume4/fcat11          Faster "cat" for Gulam, v 1.1
volume4/popen      popen(3) call for plain TOS
volume4/eplay20         STe sound sample player
volume4/etext      Filter ASCII files into PageStream format
```

```
volume4/gulmake         Files for compiling gcc under Gulam
volume4/gnum4/part01        GNU "m4" macro processor, v 0.50
volume4/gnum4/part02
volume4/gnum4/part03
volume4/gnum4/part04
volume4/gnum4/part05
volume4/mkrsc/part01        Resource construction program
volume4/mkrsc/part02
volume4/mkrsc/part03
volume4/cdecl-p1        Generate C declarations, patch 1
volume4/mgif       GIF viewer for monochrome STs
volume4/gawk211/part01      GNU awk, version 2.11
volume4/gawk211/part02
volume4/gawk211/part03
volume4/gawk211/part04
volume4/gawk211/part05
volume4/gawk211/part06
volume4/gawk211/part07
volume4/motasm          Motorola assemblers for 68xx processors
volume4/xlharc12/part01     Another version of lharc (based on UNIX lharc 0.03)
volume4/xlharc12/part02
volume4/sozosun         Diffs to Sozobon C for cross-compiling on Suns
volume4/zlib       Library functions for reading .Z files
volume4/killdrv         Add/remove drives from TOS's drive list
volume4/szadb12/part01      Sozobon adb debugger, v 1.2
volume4/szadb12/part02
volume4/szadb12/part03
volume4/pml_sqrt        Fast square root function for gcc
volume4/gemfstfx        Fixes for vdifast from Gemfast 1.3
volume4/xstrip          eXtended Strip, for removing symbol tables
volume4/lynxlib/part01      Library of miscellaneous routines
volume4/lynxlib/part02
volume4/lynxlib/part03
volume4/lynxlib/part04
volume4/mclock          Corner clock for mono systems
volume4/mgif11          GIF viewer for mono systems, v 1.1
volume4/nlight          Show I/O to any disk by flashing floppy light
volume4/less97/part01       file pager for MiNT
volume4/less97/part02
volume4/less97/part03
volume4/less97/part04
volume4/pathalias/part01        Build UUCP paths file from maps
volume4/pathalias/part02
volume4/pathalias/part03
volume4/dvi2tty         Plain text dvi previewer
volume4/mercury/part01      Mercury UUCP: mail/news package
volume4/mercury/part02
volume4/mercury/part03
volume4/mercury/part04
volume4/mercury/part05
volume4/mercury/part06
volume4/nroff110p4/part01       Nroff clone, version 1.10 patchlevel 4
volume4/nroff110p4/part02
```

```
volume4/nroff110p4/part03
volume4/nroff110p4/part04
volume4/man        Read online manual pages
volume4/manpg14/part01      Pager for reading online manual pages
volume4/manpg14/part02
volume4/manpg14/part03
volume4/dvithead       dvi2tty.h (missing file from dvi2tty)
volume4/diff115/part01      GNU diff, version 1.15
volume4/diff115/part02
volume4/diff115/part03
volume4/diff115/part04
volume4/rcs5ap1/part01      Revision Control System, version 5
volume4/rcs5ap1/part02
volume4/rcs5ap1/part03
volume4/rcs5ap1/part04
volume4/rcs5ap1/part05
volume4/rcs5ap1/part06
volume4/rcs5ap1/part07
volume4/rcs5ap1/part08

-- VOLUME 5 --

volume5/geniscan       Software to read from GeniScan scanner
volume5/tee        Copy stdin to stdout as well as a file
volume5/orctool         Clones of various UNIX utilities
volume5/tr         Transform characters (UNIX utility clone)
volume5/wc         wc(1) character/word/line counting filter
volume5/chedit          Font editor for mono 8x16 fonts
volume5/regexp          Regular expression library
volume5/mntutil8/part01     Utilities for use under MiNT
volume5/mntutil8/part02
volume5/qroff      Simple text formatter
volume5/loadspx         Spectrum picture viewer
volume5/bauble/part01       Bauble UUCP utilities (MiNT-friendly)
volume5/bauble/part02
volume5/bauble/part03
volume5/brik/part01         Generate/check CRCs on files
volume5/brik/part02
volume5/brik/part03
volume5/nasm/part01         6502 assembler/linker package
volume5/nasm/part02
volume5/nasm/part03
volume5/nasm/part04
volume5/nasm/part05
volume5/crabs           Screen-eating program for MiNT
volume5/con             Redirect MiNT status info to a window under MGR
volume5/mgif32/part01       Monochrome GIF viewer
volume5/mgif32/part02
volume5/speed           Switch between 8 and 16MHz
volume5/showgs              Preview GhostScript printer files onscreen
volume5/xxxx_it             Easily choose MiNT, Spectre, or GEM desktop at boot
volume5/uucoder             GEM-based uuencode/decode utility
volume5/bituue              Uuencode a file for mailing across BITNET
```

```
volume5/zoo21/part01        Version 2.1 of Zoo archiver
volume5/zoo21/part02
volume5/zoo21/part03
volume5/zoo21/part04
volume5/zoo21/part05
volume5/zoo21/part06
volume5/zoo21/part07
volume5/zoo21/part08
volume5/zoo21/part09
volume5/zoo21/part10
volume5/lynxlib2/part01     Library of miscellaneous routinves, v2
volume5/lynxlib2/part02
volume5/lynxlib2/part03
volume5/lynxlib2/part04
volume5/sokoban2        Game: push blocks around the board
volume5/stmenu              Popup menu library
volume5/ff32/part01         Floppy disk speedup program
volume5/ff32/part02
volume5/arc2fol             Bulk unarchiver (unpack into separate dirs)
volume5/shuffle             Game: use a picture as a puzzle
volume5/oridiff             Diffs to origami folding editor source
volume5/mint091/part01      Multitasking TOS replacement, 0.91
volume5/mint091/part02
volume5/mint091/part03
volume5/mint091/part04
volume5/mint091/part05
volume5/mint091/part06
volume5/mint091/part07
volume5/mint091/part08
volume5/mint091/part09
volume5/mint091/part10
volume5/tqgem100/part01     GEM bindings for FTL Modula 2
volume5/tqgem100/part02
volume5/tqgem100/part03
volume5/tqgem100/part04
volume5/lynxup              Update to lynxlib2
volume5/hpdjplus        First Word Plus driver file for HP DeskJet
volume5/sozobon2/part01     Sozobon C compiler, version 2
volume5/sozobon2/part02
volume5/sozobon2/part03
volume5/sozobon2/part04
volume5/sozobon2/part05
volume5/sozobon2/part06
volume5/sozobon2/part07
volume5/szadb21/part01      Debugger for Sozobon C
volume5/szadb21/part02
volume5/szadb21/part03
volume5/bootx           Configure system at boot time
volume5/defdfn              Translate files between DR RCS 1.x and 2.0
volume5/rsctoc              Convert RSC files to C source
volume5/mclock2             100-hour clock for mono systems, v2
volume5/emacs.rc        Micro-EMACS startup file for C source editing
volume5/booted17        Boot-time system configuration editor
```

```
volume5/hdwait3             Wait for hard disk to spin up, v3
volume5/double2             Double vertical screen resolution on STEs, v2
volume5/gccregs             Patches to MINIX gcc 1.40 for register arguments
volume5/tapebios/part01     Access SCSI streaming tape drives
volume5/tapebios/part02
volume5/tapebios/part03
volume5/scsitape       SCSI tape driver + utilities
volume5/texinfo2.14/part01  Process GNU online documentation
volume5/texinfo2.14/part02
volume5/texinfo2.14/part03
volume5/texinfo2.14/part04
volume5/scsitape2      SCSI driver + utilities, version 2
volume5/rdsh           Utilities for rde_v2 RAMdisk
volume5/sh/part01      Bourne shell implementation for MiNT
volume5/sh/part02
volume5/sh/part03
volume5/sh/part04
volume5/sh/part05
volume5/sh/part06
volume5/sh/part07


From 8147504806322331188
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,84f9a0809132dedc,start
X-Google-Attributes: gid100493,public
X-Google-Thread: 111a55,84f9a0809132dedc,start
X-Google-Attributes: gid111a55,public
X-Google-ArrivalTime: 1992-06-29 13:06:54 PST
Xref: sparky comp.sources.atari.st:53 comp.sys.atari.st:10667
Path: sparky!uunet!sun-barr!news2me.ebay.sun.com!exodus.Eng.Sun.COM!
appserv.Eng.Sun.COM!appserv.Eng.Sun.COM!news
From: koreth@twitterpater.eng.sun.com (Steven Grimm)
Newsgroups: comp.sources.atari.st,comp.sys.atari.st
Subject: v06INF1: Introduction to comp.sources.atari.st
Date: 30 Jun 1992 01:00:47 GMT
Organization: Sun Microsystems Inc., Mountain View, CA, USA
Lines: 154
Approved: koreth@twitterpater.eng.sun.com (Steven Grimm)
Expires: 120days
Message-ID: <l4vchvINNt3s@appserv.Eng.Sun.COM>
NNTP-Posting-Host: twitterpater

Submitted-by: koreth@twitterpater.eng.sun.com (Steven Grimm)
Posting-number: Volume 6, Info 1
Archive-name: intro

This is the first of two introductory articles about comp.sources.atari.st.
This one describes how to submit source to the newsgroup, where the
archive sites are, and how to contact them.  The companion article lists
all previously-published sources.

I am always looking for suggestions on how to improve the usefulness
```

of the newsgroup, and can be contacted as listed below.

-- Steven Grimm
   koreth@twitterpater.eng.sun.com

Subject: Submitting source for publication

Items intended for posting or queries and problem notes should be sent to atari-sources@twitterpater.eng.sun.com. If you are on a UUCP-only site, you can send them to sun!twitterpater.eng!atari-sources.

If you want verification of arrival, so say in a cover note, or at the beginning of your submission, if it is small. I try to verify that a program works, and if I can't get it to work, I may hold up posting it for a couple of days. Please note that, except in rare cases, source without documentation will not be published. The backlog from receipt to posting varies from one to four weeks depending mostly on the set of submissions currently in my queue.

I prefer that you submit sources and binaries separately, as it's a pain to separate them, and such submissions are invariably placed on the back burner. Sources should be submitted in the same format as binaries -- uuencoded arc, zoo, or lzh files. Please use Pierre Dumas' "uue" program if possible, as it can correct some common transmission errors. Split long submissions up into reasonable (less than 50K) chunks. "uue" will do this for you, if you use "uue -700 file" to encode.

Please inform me of any special system requirements, and what development environment was used, so that I can properly test your submission and label it when I post it.

Subject: The structure of comp.sources.atari.st articles

Each posting in comp.sources.atari.st is called an "issue"; there are roughly 100 issues to a volume. The division is arbitrary and may vary. There are two types of articles in comp.sources.atari.st: sources and "information postings." They can be distinguished by the subject line:

  Subject: v06INF2: Index for Volume 6 and other info

This first word in the title identifies this as the third info posting of volume one. Similarly, the subject line shown below:

  Subject: v06i081: godsrc -- Universe simulator

identifies this as the 81st source article in Volume 6. Large sources are broken up into smaller pieces, and have subject lines that look like this:

```
  Subject: v06i088: decbustr -- VMS clone in ST-LOGO, Part03/08
```

Certain information about the system configuration required to use the program is given on the keywords line.

```
  Keywords: MWC, Megamax, 1meg, high
```

This means that the program compiles with Mark Williams or Megamax C, requires at least one meg of RAM and only runs in high resolution. Following is a list of keywords; new ones may be added as needed. They are mostly self-explanatory.

```
  Alcyon  - Alcyon C (included with Atari developer's kit)
  MWC         - Mark Williams C
  Megamax - Megamax C
  Laser       - Laser C
  Lattice - Lattice C
  C       - Any C compiler
  OSS         - OSS Personal Pascal
  BASIC       - ST BASIC (this is not a joke!)
  uuencode    - program is uuencoded (uudecode required to unpack)
  arc         - program is archived (arc required to unpack)
  zoo-high    - program is archived (Zoo 2.1 required to unpack)
  lharc       - program is archived (lharc required to unpack)
  high        - high resolution
  medium  - medium resolution
  low         - low resolution
  1meg        - needs 1 meg of RAM
```

The References: line, if present, points to the previous part of a multi-part program, so that newsreaders (trn, nn, etc.) which can follow threads will recognize a large program as a single thread.

The first few lines of an article are auxiliary headers that look like this:

```
  Submitted-by: david@bdt.UUCP (David Beckmeyer)
  Posting-number: Volume 6, Issue 80
  Archive-name: micro-rtx
```

The "Submitted by" is the author of the program. If you have comments about the sources published in comp.sources.atari.st this is the person to contact. When possible, this address is in domain form, otherwise it is a UUCP bang path relative to some major (backbone) site.

The second line repeats the volume/issue information for the aide of notes sites and automatic archiving programs.

The Archive-name is the "official" name of this source in the archive. Large postings will have names that look like this:

```
Archive-name: desktop/part01
```

Please try to use this name when requesting that sources be mailed to you.
Also, note that the "part number" given in the title, and the archive name
given in the auxiliary header need not be identical.

Since most archive sites run UNIX, articles are given UNIX-style filenames
rather than ST-style filenames. I do try to limit filenames to eight
characters whenever possible, though.

Subject: Reporting and tracking bugs and patches to postings

Most of the time, bugs and updates will be posted to comp.sys.atari.st.
Sometimes, "official" updates from the author will be posted submitted to
comp.sources.atari.st so that they are archived with the sources.

Since there is no easy way to apply source code patches on the ST, changes
to a program often result in a whole new version being posted.

To report bugs, contact the person listed in the Submitted-to header.
Often there is a contact address in a README file, too. I do not maintain
the sources I moderate, so don't send your bug reports to me.

Subject: Listing of archive sites in no particular order

1. Site twitterpater is running an archive server, which automatically answers
   mail requests. Send mail to archive-server@twitterpater.eng.sun.com with the
   word "help" in the message body for more information. Both the sources
   and binaries groups are archived on twitterpater.
2. Unido is the European archive site. It's running an archive server
   similar to twitterpater's; send "help" to unido!archive-server for more
   information. Unido archives both the sources and binaries groups.

If you know of additional sites which reliably archive comp.sources.atari.st
(or comp.binaries.atari.st, for that matter), please tell me so I can add
them to the list.

From -5963676429021302228
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,d08c26eaf319a29d,start
X-Google-Attributes: gid111a55,public
X-Google-Thread: 156df7,d08c26eaf319a29d,start
X-Google-Attributes: gid156df7,public
X-Google-Thread: 100493,d08c26eaf319a29d,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1992-07-07 03:55:54 PST
Xref: sparky comp.sources.atari.st:61 comp.binaries.atari.st:163
comp.sys.atari.st:10957
Path: sparky!uunet!zaphod.mps.ohio-state.edu!cs.utexas.edu!sun-barr!
news2me.ebay.sun.com!exodus.Eng.Sun.COM!appserv.Eng.Sun.COM!

twitterpater.Eng.Sun.COM!koreth
From: koreth@twitterpater.Eng.Sun.COM (Steven Grimm)
Newsgroups: comp.sources.atari.st,comp.binaries.atari.st,comp.sys.atari.st
Subject: INFO: All submissions have been posted (finally)
Followup-To: comp.sys.atari.st
Date: 7 Jul 1992 15:47:40 GMT
Organization: The Central Stupidity Agency, Washington DC, USA
Lines: 48
Approved: koreth@twitterpater.Eng.Sun.COM (Steven Grimm)
Message-ID: <l5jf4sINN1rn@appserv.Eng.Sun.COM>
NNTP-Posting-Host: twitterpater

Thanks to a nice vacation last week, the comp.binaries.atari.st and comp.sources.atari.st submission queues are now empty. Time to send in some more!

The rest of this message will sound a lot like a similar one in comp.sources.unix not long ago. You can stop reading here if you don't plan to submit things to either group.

My response time is usually nothing to write home about. Part of the reason is that I try to send things out in more or less the order they arrive (with some exceptions.) When I get a slew of three or four submissions that have problems (both source and binary included in one archive, weird archiver used to pack, whatever) they tend to sit at the head of the queue until I have time to deal with them. Perfectly clean submissions further back in the queue suffer as a result, and it takes a big chunk of free time like last week to get everything moving again.

So, in the name of keeping the trains running on time, I'm going to be a little stricter about rejecting submissions in the future. Up until now, the only reasons I rejected things were if I couldn't get them to unpack at all (usually due to data corruption) or if I tried them out and they didn't work. I'll still reject things for those reasons, of course, but I'll no longer spend time going through the zillions of different versions of LHarc one at a time to try to unpack something, trying to figure out which of the datafiles need to be included with both the source version and the binary version of something that arrived all in one chunk, or searching through the source code for a big comment in the middle that can be used as a description for the top of the article. If it's not pretty much ready to go when I get it, I'll tell you what's wrong with it and move to the next item. I'm still perfectly willing to repackage things with zoo if they come in packaged with arc (I feel the space savings, sometimes in excess of 30%, are worth it) or whatever -- but I'll limit myself to fiddling with the format of the submission in a coarse way, rather than shuffling through its contents.

There's nothing really new here; I'm not asking anything that hasn't been in the two groups' introductory articles pretty much from the start. Most submitters will be perfectly fine; in many cases, I don't have to touch incoming submissions at all, except to change the mail header to a news header. I don't expect to be able to do that all the

time, or even most of the time, but I'd like to stop the practice of spending three hours trying to get a single submission into a form people can use.

Feel free to post or mail comments or replies.

-Steve

From 4290938461937267081
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,70986401fccf30ec,start
X-Google-Attributes: gid100493,public
X-Google-Thread: 156df7,70986401fccf30ec,start
X-Google-Attributes: gid156df7,public
X-Google-ArrivalTime: 1992-09-02 02:37:58 PST
Xref: sparky comp.sources.atari.st:63 comp.binaries.atari.st:174
Path: sparky!uunet!cs.utexas.edu!usc!elroy.jpl.nasa.gov!ames!sun-barr!
west.West.Sun.COM!grapevine.EBay.Sun.COM!grapevine.EBay.Sun.COM!news
From: koreth@twitterpater.Eng.Sun.COM (Steven Grimm)
Newsgroups: comp.sources.atari.st,comp.binaries.atari.st
Subject: ADMIN: Submission addresses changing
Date: 2 Sep 1992 08:27:25 -0700
Organization: Sun Microsystems Federal, Milpitas, CA.
Lines: 17
Sender: sgrimm@grapevine.EBay.Sun.COM
Approved: koreth@twitterpater.Eng.Sun.COM (Steven Grimm)
Message-ID: <182mgtINNoni@grapevine.EBay.Sun.COM>
NNTP-Posting-Host: grapevine.ebay.sun.com

Effective immediately, the submission address for comp.sources.atari.st is

atari-sources@hyperion.com

and the submission address for comp.binaries.atari.st is

atari-binaries@hyperion.com

The archive server will remain on twitterpater a little while longer, but it will be moving to hyperion.com shortly as well. I will post another note when that's about to happen.

Incidentally, for those wondering why both groups are dry, it's not because I've gotten slow again; my queues have been empty for a while, and no new submissions have been coming in.

-Steve

From 1435317678164198421
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 111a55,b23ef7de6ee4e84b,start
X-Google-Attributes: gid111a55,public

X-Google-Thread: 100493,b23ef7de6ee4e84b,start
X-Google-Attributes: gid100493,public
X-Google-Thread: 156df7,b23ef7de6ee4e84b,start
X-Google-Attributes: gid156df7,public
X-Google-ArrivalTime: 1992-09-15 02:13:28 PST
Xref: sparky comp.sys.atari.st:13822 comp.binaries.atari.st:175
comp.sources.atari.st:64
Path: sparky!uunet!decwrl!netcomsv!spud.Hyperion.COM!spud.Hyperion.COM!not-for-
mail
From: koreth@spud.Hyperion.COM (Steven Grimm)
Newsgroups: comp.sys.atari.st,comp.binaries.atari.st,comp.sources.atari.st
Subject: INFO: Twitterpater becomes spud
Date: 15 Sep 1992 09:25:05 -0700
Organization: Hyperion, Mountain View, CA, USA
Lines: 11
Approved: koreth@spud.Hyperion.COM
Message-ID: <1952p1INNrka@spud.Hyperion.COM>
NNTP-Posting-Host: spud.hyperion.com
Summary: archive-server@spud.hyperion.com

Sorry for the interruption in service. I now have the archive server running
on host spud.Hyperion.COM. As on twitterpater, the archive contains all the
software from comp.sources.atari.st and comp.binaries.atari.st since I started
moderating both groups. Ftp access to the archives may be available in the
near future; more information on that as it happens.

Send mail to archive-server@spud.hyperion.com with "help" as the subject for
more information. Mail me at archive-manager@spud.hyperion.com if you have
trouble (but not to request files -- archive-server is the address for that.)

-Steve

From -4228652661258922277
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,4109d377ebf1fe83,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1987-03-17 23:39:41 PST
Relay-Version: version B 2.10 5/3/83; site utzoo.UUCP
Path: utzoo!utgpu!water!watnot!watmath!clyde!cbatt!ihnp4!ptsfa!lll-lcc!styx!ames!
ucbcad!ucbvax!decvax!decwrl!sun!imagen!atari-sources-request
From: atari-sources-request@imagen.UUCP
Newsgroups: comp.sources.atari.st
Subject: v00INF1: Welcome to comp.sources.atari.st
Message-ID: <918@imagen.UUCP>
Date: Mon, 2-Mar-87 07:04:40 EST
Article-I.D.: imagen.918
Posted: Mon Mar 2 07:04:40 1987
Date-Received: Tue, 3-Mar-87 22:52:20 EST
Sender: turner@imagen.UUCP
Lines: 152
Approved: imagen!turner

Submitted by: turner (D'arc Angel)
Comp.sources.atari.st: Volume 0, Info 1
Archive-name: welcome

Welcome to comp.sys.atari.st! This newsgroup is for the posting of
sources and documentation for the Atari ST line of computers. To
submit sources, binaries and/or documentation, send them to:

{sun,decwrl,ucbvax}!imagen!atari-sources

If you are submitting sources, please specify the compiler/assembler under
which it was compiled; for both sources and binaries please specify what
resolution(s) in which the program will run and some documentation as to
what the program does and how to use it.

Sources will be posted with a subject line of the following format:

Subject: Vxxannnn typ:res:form short description

Where:
    xx   - is a 2 digit volume number
    annnn - is a letter followed by 4 digits which will uniquely
        identify the posting for online retrival

    typ   - the type of compiler/assembler used to compile the program:
        doc - documentation only, no sources
        mwc - Marc Williams C
        mmc - Megamax C
        drc - DRI (alcyon) C
        tdi - TDI Modula-2
        bas - Atari basic
        ltc - Lattice C
        dra - DRI (Alcyon) assembler
        pps - Personal Pascal
    res   - resolution(s) under which this will run:
        b&w - high resolution only
        med - medium resolution color only
        low - low resolution color only
        col - both med and low
        all - all resolutions
    form - one or more of the following seperated by semi-colons ';'
        these fields describe the order of programs that must
        be used to 'unpack' the posting:
        asc - ascii file, no unpacking necessary
        uue - use Dumas's uudecode (or Un*x uudecode)
        arc - use arc
        shr - use shar

For Example the subject line:

V01a0001 mwc:col:uue;arc desk acc to change screen color

would be a C program compilable under Marc Williams C which runs in all color resolutions; to 'unpack' it, the user would first use uudecode to generate an arc file, then use arc to extract the program, rsc files, documentation etc.

The idea behind the rather complex subject line is 2 fold; first you can see at a glance if the posting is usable on your system and second with a little work you can write a program which parses the subject line and unpacks the posting automatically.

All programs posted will be compiled and tested before they are posted.

There will be a facility for retrival of postings via email analogous to the BITNET online retrival system, more on that as soon as I get it working.

I'm sure that I've left out some compiler types or haven't covered everything, so feel free to send me email.

I think i've gone on long enough, send me some programs !!!

Subject: v00INF2: Welcome to comp.sources.atari.st
Newsgroups: comp.sources.atari.st
Approved: imagen!turner

Submitted by:
Comp.sources.atari.st: Volume 0, Info 2
Archive-name: Info

Welcome to comp.sys.atari.st! This newsgroup is for the posting of sources and documentation for the Atari ST line of computers. To submit sources, binaries and/or documentation, send them to:

{sun,decwrl,ucbvax}!imagen!atari-sources

If you are submitting sources, please specify the compiler/assembler under which it was compiled; for both sources and binaries please specify what resolution(s) in which the program will run and some documentation as to what the program does and how to use it.

Sources will be posted with a subject line of the following format:

Subject: Vxxannnn typ:res:form short description

Where:
     xx   - is a 2 digit volume number
     annnn - is a letter followed by 4 digits which will uniquely
          identify the posting for online retrival

     typ   - the type of compiler/assembler used to compile the program:

```
          doc - documentation only, no sources
          mwc - Marc Williams C
          mmc - Megamax C
          drc - DRI (alcyon) C
          tdi - TDI Modula-2
          bas - Atari basic
          ltc - Lattice C
          dra - DRI (Alcyon) assembler
          pps - Personal Pascal
     res   - resolution(s) under which this will run:
          b&w - high resolution only
          med - medium resolution color only
          low - low resolution color only
          col - both med and low
          all - all resolutions
     form  - one or more of the following seperated by semi-colons ';'
          these fields describe the order of programs that must
          be used to 'unpack' the posting:
          asc - ascii file, no unpacking necessary
          uue - use Dumas's uudecode (or Un*x uudecode)
          arc - use arc
          shr - use shar

For Example the subject line:

V01a0001 mwc:col:uue;arc desk acc to change screen color

would be a C program compilable under Marc Williams C which runs in all
color resolutions; to 'unpack' it, the user would first use uudecode to
generate an arc file, then use arc to extract the program, rsc files,
documentation etc.

The idea behind the rather complex subject line is 2 fold; first you can
see at a glance if the posting is usable on your system and second with
a little work you can write a program which parses the subject line and
unpacks the posting automatically.

All programs posted will be compiled and tested before they are posted.

There will be a facility for retrival of postings via email analogous to
the BITNET online retrival system, more on that as soon as I get it working.

I'm sure that I've left out some compiler types or haven't covered everything,
so feel free to send me email.

I think i've gone on long enough, send me some programs !!!


From 7081814175086870098
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,3ec0d0f69dc1d74c,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1987-04-15 22:43:21 PST
```

Relay-Version: version B 2.10 5/3/83; site utzoo.UUCP
Path: utzoo!utgpu!water!watmath!clyde!rutgers!ames!necntc!auspyr!imagen!atari-sources-request
From: atari-sources-request@imagen.UUCP
Newsgroups: comp.sources.atari.st
Subject: v00i002: mmc:all:shar CKSM - checksum program
Message-ID: <1086@imagen.UUCP>
Date: Thu, 16-Apr-87 00:43:21 EST
Article-I.D.: imagen.1086
Posted: Thu Apr 16 00:43:21 1987
Date-Received: Fri, 17-Apr-87 23:53:25 EST
Sender: turner@imagen.UUCP
Lines: 176
Approved: imagen!turner

Submitted by: turner (D'arc Angel)
comp.sources.atari.st: Volume 0, Issue 2
Archive-name: cksm

+=+=+=

```
#    cksblk.c
#    cksm.c
# This archive created: Wed Apr 15 21:02:20 1987
# By:    D'arc Angel (The Houses of the Holy)
export PATH; PATH=/bin:/usr/bin:$PATH
echo shar: "extracting 'cksblk.c'" '(802 characters)'
if test -f 'cksblk.c'
then
     echo shar: "will not over-write existing file 'cksblk.c'"
else
sed 's/^ X//' << \SHAR_EOF > 'cksblk.c'
     X
     Xint cksblk(block, numbyt, initck, nlines) /*  Checksum a block of bytes  */
     Xchar *block;
     Xlong int numbyt;
     Xint initck, *nlines;
     X
     X/*  Arguments:  block:  Pointer to first byte of block to be checksummed.
*/
     X/*              numbyt: Number of bytes in the block.
                 */
     X/*              initck: Initial value for checksum (i.e., partial sum
        */
     X/*           carried forward from previous blocks).
             */
     X
     X{
     X    register char *curbyt, *endptr;
     X    register int accsm;
     X
     X    endptr = block + numbyt;    /*  Pointer to end+1 of block
     */
     X    accsm = initck;             /*  Checksum accumulator.
     */
     X
     X
     X    for ( curbyt = block;  curbyt < endptr;  curbyt++ )  {
     X#ifdef vax
     X         if(*curbyt == 0x0a)
     X              (*nlines)++;
     X#endif
     X         accsm = (accsm + accsm + accsm) + (*curbyt & 0377);
     X         accsm = (accsm & 07777) + ((accsm >> 12) & 7);
     X    }
     X
     X    return(accsm);
     X}
SHAR_EOF
if test 802 -ne "`wc -c < 'cksblk.c'`"
then
     echo shar: "error transmitting 'cksblk.c'" '(should have been 802
characters)'
fi
```

```
fi
echo shar: "extracting 'cksm.c'" '(1422 characters)'
if test -f 'cksm.c'
then
     echo shar: "will not over-write existing file 'cksm.c'"
else
sed 's/^ X//' << \SHAR_EOF > 'cksm.c'
     X#include <stdio.h>
     X#ifdef vax
     X#include <fcntl.h>
     X#else
     X#include <osbind.h>
     X#endif
     X
     X#define BUFRSIZ   1024
     X#define FD_STDIN 0
     X
     Xchar buff[BUFRSIZ];
     X
     Xmain(argc, argv)  /*  cksm:  Checksum files   */
     Xint argc;
     Xchar *argv[];
     X
     X{
     X    /*  Compilation instructions:  cc -O cksm.c cksblk.c
          */
     X
     X    int fildes;
     X
     X    if (argc == 1)  {  /*  no args, so process std. input
     */
     X         chksum(FD_STDIN, "standard input");
     X    }
     X    else  {
     X         while (--argc > 0)  {
     X              if ((fildes =
     X#ifdef vax
     X                   open(*++argv, O_RDONLY | O_NDELAY)) <= 0) {
     X#else
     X                   Fopen(*++argv, 0)) <= 0) {
     X#endif
     X                  printf("cksm: can't open '%s' error no = %d\n", *argv,
fildes);
     X              }
     X              else  {
     X                   chksum(fildes, *argv);
     X                   close(fildes);
     X              }
     X         }
     X    }
     X    printf("Hit return to continue\n");
     X    scanf("%c",buff);
     X}
```

```
     X
     Xchksum(fildes, fname) /*  Read file, count bytes, compute checksum.
     */
     Xint fildes;
     Xchar *fname;
     X
     X{
     X    int csum, nlines;
     X    long bcnt, bgot;
     X    extern char buff[BUFRSIZ];
     X    extern errno;
     X
     X    csum = 0;
     X    bcnt = 0;
     X    nlines = 0;
     X
     X#ifdef vax
     X    while ((bgot = read(fildes, buff, BUFRSIZ)) > 0)  {
     X#else
     X    while ((bgot = Fread(fildes, (long)BUFRSIZ, buff)) > 0)  {
     X#endif
     X         bcnt += bgot;
     X         csum = cksblk(buff, bgot, csum, &nlines);
     X    }
     X
     X    if (bgot < 0)  {
     X         printf("cksm: error %ld reading from %s\n", bgot, fname);
     X    }
     X
     X#ifdef vax
     X    printf("bytes =%9ld(%9ld)   cksm =%5d   %s\n", bcnt, bcnt+nlines, csum,
fname);
     X#else
     X    printf("bytes =%9ld   cksm =%5d   %s\n", bcnt, csum, fname);
     X#endif
     X}
SHAR_EOF
if test 1422 -ne "`wc -c < 'cksm.c'`"
then
     echo shar: "error transmitting 'cksm.c'" '(should have been 1422 characters)'
fi
fi
exit 0
#    End of shell archive


From -2043965245567342904
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,e93cc8c6f6c46be2,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1987-04-21 16:54:53 PST
Relay-Version: version B 2.10 5/3/83; site utzoo.UUCP
Path: utzoo!mnetor!seismo!rutgers!mit-eddie!uw-beaver!tektronix!cae780!amdcad!sun!
```

imagen!atari-sources-request
From: atari-sources-request@imagen.UUCP
Newsgroups: comp.sources.atari.st
Subject: v00i003: ::shar - UW sources and man page for BSD 4.{23}
Message-ID: <1138@imagen.UUCP>
Date: Tue, 21-Apr-87 18:54:53 EST
Article-I.D.: imagen.1138
Posted: Tue Apr 21 18:54:53 1987
Date-Received: Thu, 23-Apr-87 05:03:28 EST
Sender: turner@imagen.UUCP
Lines: 1111
Approved: imagen!turner

Submitted by: ucbvax!cwruecmp!bammi (Jwahar R. Bammi)
comp.sources.atari.st: Volume 0, Issue 3
Archive-name: uwvax

+=+=+=

```
echo shar: extracting "'uw.1'" '(1732 characters)'
if test -f 'uw.1'
then
     echo shar: over-writing existing file "'uw.1'"
fi
sed 's/^X//' << \SHAR_EOF > 'uw.1'
X.TH UW local "13 November 1985"
X.UC 4
X.SH NAME
Xuw \- unix windowing system for Macintosh
X.SH SYNOPSIS
X.B uw
X.br
X.B uwtool
X[ process ] [ arg1 arg2 ...  ]
X.SH DESCRIPTION
X.I Uw
Xis a server program on unix that works with the program
X.I uw
Xon the Macintosh.  It creates a pseudo-terminal for each window,
Xgiving the user up to seven `terminals' to work with.  The host
Xprogram multiplexes the output onto one RS\-232 line.
X.PP
X.I Uw
X(on the unix system) with
X.I uw
Xon the Macintosh emulate a Lear Siegler ADM-31 terminal (tset adm31), a
XTektronix 4010, and a DEC VT52.
XEach window (on the Macintosh) has its own terminal emulation and can be
Xresized at will; the program
Xkeeps an 80x24 line display in memory for each window, but only shows the
Xlower left portion possible.
X.PP
X.I Uwtool
Xcan be executed from unix while running
X.I uw
Xto create a new window/`terminal.'  If no arguments are given, it creates
Xa new `terminal' running csh.  If a process is named, it will create a new
Xwindow with that process running in it, and when that process is terminated
Xthe window will disappear.  (i.e. `uwtool vi foo' will create a new window
Xwith vi, editing the file foo, but the window will go away when vi is exited)
XAny arguements after the process name are passed as arguements to that process.
X.SH LIMITATIONS
X.I Uw
Xand
X.I uwtool
Xare of no use on unix unless
X.I uw
Xis being run on the Macintosh.
X.br
XIf there is a stream of output in one window there will be lag in
Xrecognizing characters typed in another.
X.SH SEE ALSO
```

```
X.I uw
Xdocumentation distributed with the Macintosh program.
X.SH AUTHOR
XProgram written by John Bruner, Lawrence Livermore Laboratories 7/85
X.br
XDocument edited by Chris Borton, UC San Diego 11/13/85
SHAR_EOF
if test 1732 -ne "`wc -c 'uw.1'`"
then
     echo shar: error transmitting "'uw.1'" '(should have been 1732 characters)'
fi
echo shar: extracting "'uw.c'" '(17040 characters)'
if test -f 'uw.c'
then
     echo shar: over-writing existing file "'uw.c'"
fi
sed 's/^X//' << \SHAR_EOF > 'uw.c'
X/*
X *  uw - UNIX windows program for the Macintosh (VAX end)
X *
X * Copyright 1985 by John D. Bruner.  All rights reserved.  Permission to
X * copy this program is given provided that the copy is not sold and that
X * this copyright notice is included.
X *
X * Compile: cc -o uw -O uw.c
X */
X
X#include <sys/types.h>
X#include <sys/stat.h>
X#include <sys/socket.h>
X#include <sys/ioctl.h>
X#include <sys/wait.h>
X#include <sys/time.h>
X#include <sys/resource.h>
X#include <sys/uio.h>
X#include <signal.h>
X#include <setjmp.h>
X#include <errno.h>
X#include <fcntl.h>
X#include <stdio.h>
X#include "uw.h"
X
X#define  MAXENV   128           /* maximum environment size */
X
X#define  W_IN 0
X#define  W_OUT    1
X
X#define  NFDS 20            /* max number of file descriptors */
X
X#define  XON  0021          /* ASCII XON */
X#define  XOFF 0023          /* ASCII XOFF */
X#define  RUB  0177          /* ASCII RUBOUT */
X
```

```
X#define  META 0200          /* "meta" bit for whatever it's worth */
X
X#define  RCV_OK   (-1)          /* see recvcmd() */
X#define  RCV_META (-2)          /* see recvcmd() */
X
Xtypedef int fildes_t;           /* file descriptor data type */
Xtypedef int nwin_t;             /* window index data type */
X
Xstruct window {
X    fildes_t  w_fd;
X    char      w_tty[32];
X};
X
Xstruct window window[NWINDOW];      /* window data structures */
Xstruct window *fdmap[NFDS];          /* mapping from fd's to windows */
Xstruct window *curwin[2];       /* current input and output windows */
X
Xchar portal[] = "/tmp/uwXXXXXX";    /* UNIX-domain network address */
X
X
X/* The following are added to the environment of all child processes */
X
Xchar env_uwin[64] = "UWIN=";
Xchar *env[] = {
X    env_uwin,
X    "TERM=adm31",
X    "TERMCAP=adm31:cr=^M:do=^J:nl=^J:al=\\EE:am:le=^H:bs:ce=\\ET:cm=\\E=%+ %
+ :cl=^Z:cd=\\EY:co#80:dc=\\EW:dl=\\ER:ei=\\Er:ho=^^:im=\
\Eq:li#24:mi:nd=^L:up=^K:MT:km:",
X    NULL
X};
X
Xint ctlch[] = { -1, IAC, XON, XOFF, -1, -1, -1, -1 };  /* CTL char mapping */
X
Xstruct selmask {
X    int  sm_rd;
X    int  sm_wt;
X    int  sm_ex;
X} selmask[2];
X
Xextern char *strncpy(), *strncat();
Xextern char *mktemp();
Xextern char *getenv();
Xextern done(), cwait(), onalarm();
Xextern int errno;
X
Xmain(argc, argv)
Xchar **argv;
X{
X    register fildes_t fd, sd;
X    register struct window *w;
X    struct sockaddr sa;
X
```

```
X    /*
X     * Make sure we don't accidentally try to run this inside itself.
X     */
X    if (getenv("UWIN")) {
X        fprintf(stderr, "%s is already running\n", *argv);
X        exit(1);
X    }
X
X    /*
X     * Close all file descriptors except 0, 1, and 2.
X     */
X
X    for (fd=3; fd < NFDS; fd++)
X        (void)close(fd);
X
X    /*
X     * Mark all windows closed.
X     */
X
X    for (w=window; w < window+NWINDOW; w++)
X        w->w_fd = -1;
X
X
X    /*
X     * Create a UNIX-domain network address, and put its name into
X     * the environment so that descendents can contact us with new
X     * window requests.
X     */
X
X    (void)strncat(env_uwin, mktemp(portal), sizeof env_uwin - 1);
X    setenv(env);
X
X    if ((sd=socket(AF_UNIX, SOCK_DGRAM, 0)) < 0) {
X        perror("socket");
X        exit(1);
X    }
X    sa.sa_family = AF_UNIX;
X    (void)strncpy(sa.sa_data, portal, sizeof sa.sa_data-1);
X    sa.sa_data[sizeof sa.sa_data-1] = '\0';
X    if (bind(sd, &sa, sizeof sa.sa_family + strlen(sa.sa_data)) < 0) {
X        perror("bind");
X        exit(1);
X    }
X    if (fcntl(sd, F_SETFL, FNDELAY))
X        perror("fcntl(sd, F_SETFL, FNDELAY)");
X
X
X    /*
X     * Ignore interrupts, quits, and terminal stops.  Clean up and exit
X     * if a hangup or termination is received.  Also catch changes in
X     * child status (so that we can wait for them).  Set up the terminal
X     * modes.
X     */
```

```
X
X    (void)signal(SIGHUP, done);
X    (void)signal(SIGINT, SIG_IGN);
X    (void)signal(SIGQUIT, SIG_IGN);
X    (void)signal(SIGTERM, done);
X    (void)signal(SIGTSTP, SIG_IGN);
X    (void)signal(SIGCHLD, cwait);
X
X    tmode(1);
X
X
X    /*
X     * Tell the Macintosh to initialize.  Initialize the current input
X     * and output windows to NULL.
X     */
X
X    xmitcmd(CB_FN_MAINT|CB_MF_ENTRY);
X    curwin[W_IN] = curwin[W_OUT] = NULL;
X
X
X    /*
X     * Initialize the "select" masks, create window 1 (to start
X     * things off) and wait for input.  When input is available,
X     * process it.
X     */
X
X    selmask[0].sm_rd = (1<<0)|(1<<sd);
X    selmask[0].sm_wt = 0;
X    selmask[0].sm_ex = selmask[0].sm_rd | 2;
X    if (newwind(window) == 0)
X         xmitcmd(CB_FN_NEWW|1);
X
X    while (1) {
X         selmask[1] = selmask[0];
X         if (select(NFDS, &selmask[1].sm_rd, &selmask[1].sm_wt,
X             &selmask[1].sm_ex, (struct timeval *)0) < 0) {
X              if (errno == EINTR)
X                   continue;
X              perror("select");
X              done(1); /* for now -- fix this! */
X         }
X         for (fd=0; fd < NFDS; selmask[1].sm_rd >>= 1, fd++) {
X              if (selmask[1].sm_rd&(1<<0)) {
X                   if (fd < 2)
X                        mrecv();
X                   else if (fd == sd)
X                        netrecv(sd);
X                   else
X                        mxmit(fd);
X              }
X         }
X    }
X}
```

```
X
Xmrecv()
X{
X    register int len;
X    register char *cp, *cq;
X    auto int nready;
X    char ibuf[512], obuf[512];
X    static int seen_iac, seen_meta;
X
X    /*
X     * The received bytestream is examined.  Non-command bytes are
X     * written to the file descriptor corresponding to the current
X     * "input" window (relative to the Macintosh -- the window the
X     * user types input to).
X     */
X
X    if (ioctl(0, (int)FIONREAD, (char *)&nready) < 0) {
X         perror("FIONREAD");
X         return;
X    }
X
X    cq = obuf;
X    while (nready > 0) {
X         if (nready > sizeof ibuf)
X              len = read(0, ibuf, sizeof ibuf);
X         else
X              len = read(0, ibuf, nready);
X         if (len <= 0) {
X              perror("read");
X              return;
X         }
X         for (cp=ibuf; cp < ibuf+len; cp++) {
X              if (seen_iac) {
X                   if ((*cp&CB_DIR) == CB_DIR_MTOH) {
X                        if (cq > obuf) {
X                             (void)write(curwin[W_IN]->w_fd,
X                                      obuf, cq-obuf);
X                             cq = obuf;
X                        }
X                        switch (*cq = recvcmd(*cp)) {
X                        case RCV_OK:
X                             break;
X                        case RCV_META:
X                             seen_meta = 1;
X                             break;
X                        default:
X                             if (seen_meta) {
X                                  seen_meta = 0;
X                                  *cq |= META;
X                             }
X                             if (curwin[W_IN])
X                                  cq++;
X                             break;
```

```
X                       }
X                  }
X                  seen_iac = 0;
X             } else if (*cp == IAC)
X                  seen_iac++;
X             else if (curwin[W_IN]) {
X                  if (seen_meta) {
X                       seen_meta = 0;
X                       *cq++ = *cp|META;
X                  } else
X                       *cq++ = *cp;
X                  if (cq >= obuf+sizeof obuf) {
X                       (void)write(curwin[W_IN]->w_fd,
X                                obuf, cq-obuf);
X                       cq = obuf;
X                  }
X             }
X        }
X        nready -= len;
X    }
X    if (cq > obuf)
X         (void)write(curwin[W_IN]->w_fd, obuf, cq-obuf);
X}
X
Xrecvcmd(cmd)
Xchar cmd;
X{
X    register int nwin, fn;
X    register struct window *w;
X
X    /*
X     * Perform the function the Mac is asking for.  There are three
X     * broad categories of these functions: RCV_META, which tells
X     * the caller that the next character is a "meta" character;
X     * an ASCII data character, which is passed back to the caller
X     * for proper handling; and RCV_OK, which means that this routine
X     * has done everything which was required to process the command.
X     */
X
X    fn = cmd&CB_FN;
X    switch (fn) {
X    case CB_FN_NEWW:
X    case CB_FN_KILLW:
X    case CB_FN_ISELW:
X         nwin = cmd&CB_WINDOW;
X         if (!nwin)
X              break;
X         w = &window[nwin-1];
X
X         switch (fn) {
X         case CB_FN_NEWW:
X              if (w->w_fd < 0 && newwind(w) < 0)
X                   xmitcmd(CB_FN_KILLW|nwin);
```

```
X             break;
X
X        case CB_FN_KILLW:
X             killwind(w, 0);
X             break;
X
X        case CB_FN_ISELW:
X             if (w->w_fd >= 0)
X                  curwin[W_IN] = w;
X             break;
X        }
X        break;
X
X    case CB_FN_META:
X        return(RCV_META);
X
X    case CB_FN_CTLCH:
X        return(ctlch[cmd&CB_CC]);
X
X    case CB_FN_MAINT:
X        if ((cmd&CB_MF) == CB_MF_EXIT)
X             done(0);
X        /*NOTREACHED*/
X    }
X    return(RCV_OK);
X}
X
Xxmitcmd(cmd)
Xchar cmd;
X{
X    static char cmdbuf[2] = { IAC, '\0' };
X
X    /*
X     * Transmit the command "cmd" to the Macintosh.  The byte is ORed
X     * with the host-to-Mac direction indicator.
X     */
X
X    cmdbuf[1] = cmd|CB_DIR_HTOM;
X    (void)write(1, cmdbuf, sizeof cmdbuf);
X}
X
Xnetrecv(sd)
Xregister fildes_t sd;
X{
X    register struct window *w;
X    register int cnt;
X    struct msghdr msg;
X    auto int fd;
X    struct iovec iov;
X    struct stat st1, st2;
X    static int unity = 1;
X    char buf[256];
X
```

```
X
X    /*
X     * main() calls this routine when there is a message waiting on
X     * the UNIX-domain socket.  The message's access rights are
X     * expected to contain the file descriptor for the "master" side
X     * of a pseudo-tty.  The message contains the name of the pty.
X     * The sender is expected to start up a process on the slave side
X     * of the pty.  This allows the host end to create windows which
X     * run something other than the shell.
X     */
X
X    fd = -1;
X
X    iov.iov_base = (caddr_t)buf;
X    iov.iov_len = sizeof buf - 1;
X
X    msg.msg_name = (caddr_t)0;
X    msg.msg_namelen = 0;
X    msg.msg_iov = &iov;
X    msg.msg_iovlen = 1;
X    msg.msg_accrights = (caddr_t)&fd;
X    msg.msg_accrightslen = sizeof fd;
X
X    if ((cnt=recvmsg(sd, &msg, 0)) < 0)
X         perror("recvmsg");
X
X    if (msg.msg_accrightslen > 0 && fd >= 0) {
X         /*
X          * We can't trust the process which connected to us, so
X          * we verify that it really passed us a pseudo-tty's
X          * file descriptor by checking the device name and its
X          * inode number.  [Of course, if someone else wants to
X          * hand us a terminal session running under their uid....]
X          */
X         buf[cnt] = 0;
X         if (strncmp(buf, "/dev/ptyp", sizeof "/dev/ptyp" - 1) ||
X             fstat(fd, &st1) < 0 || stat(buf, &st2) < 0 ||
X             st1.st_dev != st2.st_dev || st1.st_ino != st2.st_ino) {
X              (void)close(fd);
X              return;
X         }
X         /*
X          * OK, we believe the sender.  We allocate a window and
X          * tell the Macintosh to create that window on its end.
X          */
X         buf[5] = 't';      /* switch to "/dev/ttyp?" */
X         for (w=window; w < window+NWINDOW; w++) {
X              if (w->w_fd < 0) {
X                   w->w_fd = fd;
X                   fdmap[fd] = w;
X                   selmask[0].sm_rd |= (1<<fd);
X                   (void)strncpy(w->w_tty, buf, sizeof w->w_tty-1);
X                   xmitcmd(CB_FN_NEWW|(w-window+1));
```

```
X                   break;
X               }
X           }
X           /*
X            * If we have no free windows, then we close the file
X            * descriptor (which will terminate the slave process).
X            */
X           if (w == window+NWINDOW)
X               (void)close(fd);
X           /*
X            * Set non-blocking I/O on the master side.
X            */
X           (void)ioctl(fd, FIONBIO, &unity);
X       }
X}
X
Xmxmit(fd)
Xregister fildes_t fd;
X{
X    register char *cp, *cq, i;
X    register int len;
X    char ibuf[16], obuf[16];
X
X    /*
X     * Copy input from file "fd" to the Macintosh.  Be sure to convert
X     * any embedded IAC characters.
X     *
X     * Note that the input/output buffers should NOT be very large.
X     * It is undesirable to perform large reads and effectively
X     * "lock out" all other file descriptors.  The chosen size
X     * should preserve a reasonable amount of efficiency.
X     */
X
X    if (fdmap[fd]) {
X         curwin[W_OUT] = fdmap[fd];
X         xmitcmd(CB_FN_OSELW|(fdmap[fd]-window+1));
X         cq = obuf;
X         if ((len = read(fd, ibuf, sizeof ibuf)) < 0 &&
X             errno != EWOULDBLOCK) {
X              killwind(fdmap[fd], 1);
X              return;
X         }
X         for (cp=ibuf; cp < ibuf+len; cp++) {
X              if (*cp&META) {
X                   if (cq > obuf) {
X                        (void)write(1, obuf, cq-obuf);
X                        cq = obuf;
X                   }
X                   xmitcmd(CB_FN_META);
X                   *cp &= ~META;
X              }
X              i = -1;
X              if (*cp == RUB || *cp < ' ') {
```

```
X                  i = sizeof ctlch;
X                  while (i >= 0 && ctlch[i] != *cp)
X                       i--;
X             }
X             if (i >= 0) {
X                  if (cq > obuf) {
X                       (void)write(1, obuf, cq-obuf);
X                       cq = obuf;
X                  }
X                  xmitcmd(CB_FN_CTLCH|i);
X             } else {
X                  *cq++ = *cp;
X                  if (cq >= obuf+sizeof obuf) {
X                       (void)write(1, obuf, cq-obuf);
X                       cq = obuf;
X                  }
X             }
X        }
X    } else
X        (void)read(fd, ibuf, sizeof ibuf);
X    if (cq > obuf)
X        (void)write(1, obuf, cq-obuf);
X}
X
Xkillwind(w, notify)
Xregister struct window *w;
X{
X    register int mask;
X
X    /*
X     * Kill window "w".  Notify the Macintosh that it is gone if
X     * "notify" is nonzero.
X     */
X
X    (void)close(w->w_fd);
X    mask = ~(1<<w->w_fd);
X    fdmap[w->w_fd] = NULL;
X    w->w_fd = -1;
X    if (curwin[W_IN] == w)
X        curwin[W_IN] = NULL;
X    if (curwin[W_OUT] == w)
X        curwin[W_OUT] = NULL;
X    selmask[0].sm_rd &= mask;
X    selmask[0].sm_wt &= mask;
X    selmask[0].sm_ex &= mask;
X    if (notify)
X        xmitcmd(CB_FN_KILLW|(w-window+1));
X}
X
Xnewwind(w)
Xregister struct window *w;
X{
X    register char *cp;
```

```
X    register fildes_t fd;
X    register int pid;
X    register char *shell;
X    char pty[32], ptysufx[2];
X    static char ptyidx[] = "0123456789abcdefghijklmnopqrstuvwxyz";
X    static int unity = 1;
X    extern char *getenv();
X
X    /*
X     * Create a new window using the specified component of "window".
X     * This routine isn't very smart at finding pseudo-ttys.
X     */
X
X    ptysufx[1] = '\0';
X    for (cp=ptyidx; *cp; cp++) {
X         ptysufx[0] = *cp;
X         (void)strncpy(pty, "/dev/ptyp", sizeof pty-1);
X         (void)strncat(pty, ptysufx, sizeof pty-1);
X         if ((fd = open(pty, 2)) >= 0)
X              break;
X    }
X    if (fd < 0)
X         return(-1);
X    (void)ioctl(fd, FIONBIO, &unity);     /* set non-blocking I/O */
X    fdmap[fd] = w;
X    w->w_fd = fd;
X    selmask[0].sm_rd |= (1<<fd);
X    (void)strncpy(w->w_tty, "/dev/ttyp", sizeof w->w_tty-1);
X    (void)strncat(w->w_tty, ptysufx, sizeof w->w_tty-1);
X
X    while ((pid=fork()) < 0)
X         sleep(5);
X    if (!pid) {
X         (void)signal(SIGHUP, SIG_DFL);
X         (void)signal(SIGINT, SIG_DFL);
X         (void)signal(SIGQUIT, SIG_DFL);
X         (void)signal(SIGTERM, SIG_DFL);
X         (void)signal(SIGTSTP, SIG_IGN);
X         (void)signal(SIGCHLD, SIG_DFL);
X         (void)ioctl(open("/dev/tty", 2), (int)TIOCNOTTY, (char *)0);
X         (void)setuid(getuid());     /* shouldn't need this */
X         if ((fd=open(w->w_tty, 2)) < 0)
X              _exit(1);
X         if (!(shell = getenv("SHELL")))
X              shell = "/bin/sh";
X         (void)dup2(fd, 0);
X         (void)dup2(0, 1);
X         (void)dup2(0, 2);
X         for (fd=3; fd < NFDS; fd++)
X              (void)close(fd);
X         tmode(0);/* HACK! */
X         execl(shell, shell, (char *)0);
X         _exit(1);
```

```
X    }
X
X    return(0);
X}
X
Xtmode(f)
X{
X    static struct sgttyb ostty, nstty;
X    static struct tchars otchars, ntchars;
X    static struct ltchars oltchars, nltchars;
X    static int olmode, nlmode;
X    static saved;
X
X    /*
X     * This routine either saves the current terminal modes and then
X     * sets up the terminal line or resets the terminal modes (depending
X     * upon the value of "f").  The terminal line is used in "cbreak"
X     * mode with all special characters except XON/XOFF disabled.  The
X     * hated (by me) LDECCTQ mode is required for the Macintosh to
X     * handle flow control properly.
X     */
X
X    if (f == 1) {
X        if (ioctl(0, (int)TIOCGETP, (char *)&ostty) < 0) {
X            perror("ioctl((int)TIOCGETP)");
X            done(1);
X        }
X        if (ioctl(0, (int)TIOCGETC, (char *)&otchars) < 0) {
X            perror("ioctl((int)TIOCGETC)");
X            done(1);
X        }
X        if (ioctl(0, (int)TIOCGLTC, (char *)&oltchars) < 0) {
X            perror("ioctl((int)TIOCGLTC)");
X            done(1);
X        }
X        if (ioctl(0, (int)TIOCLGET, (char *)&olmode) < 0) {
X            perror("ioctl((int)TIOCLGET)");
X            done(1);
X        }
X        nstty = ostty;
X        nstty.sg_erase = nstty.sg_kill = -1;
X        nstty.sg_flags |= CBREAK;
X        nstty.sg_flags &= ~(RAW|CRMOD|ECHO|LCASE|XTABS);
X        ntchars.t_intrc = ntchars.t_quitc = -1;
X        ntchars.t_eofc = ntchars.t_brkc = -1;
X        ntchars.t_startc = XON;
X        ntchars.t_stopc = XOFF;
X        nltchars.t_suspc = nltchars.t_dsuspc = -1;
X        nltchars.t_rprntc = nltchars.t_flushc = -1;
X        nltchars.t_werasc = nltchars.t_lnextc = -1;
X        nlmode = olmode | LDECCTQ;
X        if (ioctl(0, (int)TIOCSETN, (char *)&nstty) < 0) {
X            perror("ioctl((int)TIOCSETN)");
```

```
X             done(1);
X         }
X         if (ioctl(0, (int)TIOCSETC, (char *)&ntchars) < 0) {
X             perror("ioctl((int)TIOCSETC");
X             done(1);
X         }
X         if (ioctl(0, (int)TIOCSLTC, (char *)&nltchars) < 0) {
X             perror("ioctl((int)TIOCSLTC");
X             done(1);
X         }
X         if (ioctl(0, (int)TIOCLSET, (char *)&nlmode) < 0) {
X             perror("ioctl((int)TIOCLSET)");
X             done(1);
X         }
X         saved = 1;
X     } else if (saved) {
X         (void)ioctl(0, (int)TIOCSETP, (char *)&ostty);
X         (void)ioctl(0, (int)TIOCSETC, (char *)&otchars);
X         (void)ioctl(0, (int)TIOCSLTC, (char *)&oltchars);
X         (void)ioctl(0, (int)TIOCLSET, (char *)&olmode);
X     }
X}
X
Xdone(s)
X{
X    register struct window *w;
X    register fildes_t fd;
X
X    /*
X     * Clean up and exit.  It is overkill to close all of the file
X     * descriptors, but it causes no harm.  After we are sure that
X     * our UNIX-domain network connection is closed we remove the
X     * entry that it created (as a side effect) in the filesystem.
X     *
X     * We also restore the terminal modes.
X     */
X
X    /*xmitcmd(CB_FN_MAINT|CB_MF_EXIT);*/
X    for (fd=3; fd < NFDS; fd++)
X         (void)close(fd);
X    (void)unlink(portal);
X    tmode(0);
X    exit(s);
X}
X
Xcwait()
X{
X    union wait status;
X    struct rusage rusage;
X
X    /*
X     * Collect dead children.  We don't use the information that
X     * wait3() returns.  (Someday we might.)
```

```
X     */
X
X    while (wait3(&status, WNOHANG, &rusage) > 0)
X         ;
X}
X
X
Xstatic char *earray[MAXENV+1];
X
Xsetenv(env)
Xchar **env;
X{
X    register char **ep1, **ep2, *cp;
X    char **ep3;
X    extern char **environ;
X
X
X    /*
X     * Merge the set of environment strings in "env" into the
X     * environment.
X     */
X
X    /*
X     * The first time through, copy the environment from its
X     * original location to the array "earray".  This makes it a
X     * little easier to change things.
X     */
X
X    if (environ != earray){
X         ep1=environ;
X         ep2=earray;
X         while(*ep1 && ep2 <= earray+MAXENV)
X              *ep2++ = *ep1++;
X         *ep2++ = NULL;
X         environ = earray;
X    }
X
X
X    /*
X     * If "env" is non-NULL, it points to a list of new items to
X     * be added to the environment.  These replace existing items
X     * with the same name.
X     */
X
X    if (env){
X         for(ep1=env; *ep1; ep1++){
X              for(ep2=environ; *ep2; ep2++)
X                   if (!envcmp(*ep1, *ep2))
X                        break;
X              if (ep2 < earray+MAXENV) {
X                   if (!*ep2)
X                        ep2[1] = NULL;
X                   *ep2 = *ep1;
```

```
X             }
X        }
X    }
X
X
X    /* Finally, use an insertion sort to put things in order. */
X
X    for(ep1=environ+1; cp = *ep1; ep1++){
X         for(ep2=environ; ep2 < ep1; ep2++)
X              if (envcmp(*ep1, *ep2) < 0)
X                   break;
X         ep3 = ep2;
X         for(ep2=ep1; ep2 > ep3; ep2--)
X              ep2[0] = ep2[-1];
X         *ep2 = cp;
X    }
X}
X
X
Xstatic
Xenvcmp(e1, e2)
Xregister char *e1, *e2;
X{
X    register d;
X
X    do {
X         if (d = *e1 - *e2++)
X              return(d);
X    } while(*e1 && *e1++ != '=');
X    return(0);
X}
SHAR_EOF
if test 17040 -ne "`wc -c 'uw.c'`"
then
     echo shar: error transmitting "'uw.c'" '(should have been 17040 characters)'
fi
echo shar: extracting "'uw.h'" '(2697 characters)'
if test -f 'uw.h'
then
     echo shar: over-writing existing file "'uw.h'"
fi
sed 's/^X//' << \SHAR_EOF > 'uw.h'
X/*
X *  uw command bytes
X *
X * Copyright 1985 by John D. Bruner.  All rights reserved.  Permission to
X * copy this program is given provided that the copy is not sold and that
X * this copyright notice is included.
X *
X *
X * Two types of information are exchanged through the 7-bit serial line:
X * ordinary data and command bytes.  Command bytes are preceeded by
X * an IAC byte.  IAC bytes and literal XON/XOFF characters (those which
```

```
X * are not used for flow control) are sent by a CB_FN_CTLCH command.
X * Characters with the eighth bit set (the "meta" bit) are prefixed with
X * a CB_FN_META function.
X *
X * The next most-significant bit in the byte specifies the sender and
X * recipient of the command.  If this bit is clear (0), the command byte
X * was sent from the host computer to the Macintosh; if it is set (1)
X * the command byte was sent from the Macintosh to the host computer.
X * This prevents confusion in the event that the host computer
X * (incorrectly) echos a command back to the Macintosh.
X *
X * The remaining six bits are partitioned into two fields.  The low-order
X * three bits specify a window number from 1-7 (window 0 is reserved for
X * other uses) or another type of command-dependent parameter.  The next
X * three bits specify the operation to be performed by the recipient of
X * the command byte.
X *
X * Note that the choice of command bytes prevents the ASCII XON (021) and
X * XOFF (023) characters from being sent as commands.  CB_FN_ISELW commands
X * are only sent by the Macintosh (and thus are tagged with the CB_DIR_MTOH
X * bit).  Since XON and XOFF data characters are handled via CB_FN_CTLCH,
X * this allows them to be used for flow control purposes.
X */
X
X#define IAC      0001     /* interpret as command */
X#define CB_DIR        0100     /* command direction: */
X#define CB_DIR_HTOM   0000     /*   from host to Mac */
X#define CB_DIR_MTOH   0100     /*   from Mac to host */
X#define CB_FN         0070     /* function code: */
X#define CB_FN_NEWW    0000     /*   new window */
X#define CB_FN_KILLW   0010     /*   kill (delete) window */
X#define CB_FN_ISELW   0020     /*   select window for input */
X#define CB_FN_OSELW   0030     /*   select window for output */
X#define CB_FN_META    0050     /*   add meta to next data char */
X#define CB_FN_CTLCH   0060     /*   low 3 bits specify char */
X#define CB_FN_MAINT   0070     /*   maintenance functions */
X#define CB_WINDOW 0007     /* window number mask */
X#define CB_CC         0007     /* control character specifier: */
X#define CB_CC_IAC 1        /*   IAC */
X#define CB_CC_XON 2        /*   XON */
X#define CB_CC_XOFF    3        /*   XOFF */
X#define CB_MF         0007     /* maintenance functions: */
X#define CB_MF_ENTRY   0        /*   beginning execution */
X#define CB_MF_EXIT    7        /*   execution terminating */
X#define NWINDOW       7        /* maximum number of windows */
SHAR_EOF
if test 2697 -ne "`wc -c 'uw.h'`"
then
     echo shar: error transmitting "'uw.h'" '(should have been 2697 characters)'
fi
echo shar: extracting "'uwtool.c'" '(3482 characters)'
if test -f 'uwtool.c'
then
```

```
     echo shar: over-writing existing file "'uwtool.c'"
fi
sed 's/^X//' << \SHAR_EOF > 'uwtool.c'
X#
X
X/*
X *  uwtool
X *
X * Copyright 1985 by John D. Bruner.  All rights reserved.  Permission to
X * copy this program is given provided that the copy is not sold and that
X * this copyright notice is included.
X *
X * Compile: cc -o uwtool -O uwtool.c
X */
X
X#include <sys/types.h>
X#include <sys/socket.h>
X#include <sys/ioctl.h>
X#include <sys/wait.h>
X#include <sys/time.h>
X#include <sys/resource.h>
X#include <sys/uio.h>
X#include <signal.h>
X#include <stdio.h>
X
X#define  NFDS 20            /* max number of file descriptors */
X
Xtypedef int fildes_t;
X
Xextern char *strncpy(), *strncat();
Xextern char *getenv();
X
Xmain(argc, argv)
Xchar **argv;
X{
X    register char *cp;
X    register int pid;
X    register fildes_t sd;
X    auto fildes_t fd;
X    char *portal, *shell;
X    char tty[32], pty[32], ptysufx[2];
X    int lmode;
X    struct sgttyb sg;
X    struct tchars tc;
X    struct ltchars ltc;
X    struct iovec iov;
X    struct msghdr msg;
X    static char ptyidx[] = "0123456789abcdefghijklmnopqrstuvwxyz";
X    struct sockaddr sa;
X
X    /*
X     * Close all file descriptors except 0, 1, and 2.
X     */
```

```
X
X    for (fd=3; fd < NFDS; fd++)
X         (void)close(fd);
X
X    /*
X     * Get the terminal configuration for this tty.
X     */
X    (void)ioctl(0, (int)TIOCGETP, (char *)&sg);
X    (void)ioctl(0, (int)TIOCGETC, (char *)&tc);
X    (void)ioctl(0, (int)TIOCGLTC, (char *)&ltc);
X    (void)ioctl(0, (int)TIOCLGET, (char *)&lmode);
X
X    /*
X     * Create a UNIX-domain socket.
X     */
X
X    if (!(portal=getenv("UWIN"))) {
X         fprintf(stderr,
X             "You must run %s under the window manager\n", *argv);
X         exit(1);
X    }
X
X    if ((sd=socket(AF_UNIX, SOCK_DGRAM, 0)) < 0) {
X         perror("socket");
X         exit(1);
X    }
X    sa.sa_family = AF_UNIX;
X    (void)strncpy(sa.sa_data, portal, sizeof sa.sa_data-1);
X    sa.sa_data[sizeof sa.sa_data-1] = '\0';
X
X
X    /*
X     * Obtain a pseudo-tty.  This code isn't very smart about finding one.
X     */
X
X    ptysufx[1] = '\0';
X    for (cp=ptyidx; *cp; cp++) {
X         ptysufx[0] = *cp;
X         (void)strncpy(pty, "/dev/ptyp", sizeof pty-1);
X         (void)strncat(pty, ptysufx, sizeof pty-1);
X         if ((fd = open(pty, 2)) >= 0)
X              break;
X    }
X    if (fd < 0) {
X         fprintf(stderr, "Can't obtain a pseudo-tty for a new window\n");
X         exit(1);
X    }
X    (void)strncpy(tty, "/dev/ttyp", sizeof tty-1);
X    (void)strncat(tty, ptysufx, sizeof tty-1);
X
X
X    /*
X     * Fork a child process using this pseudo-tty.  Initialize the
```

```
X     * terminal modes on the pseudo-tty to match those of the parent
X     * tty.
X     */
X
X    while ((pid=fork()) < 0)
X         sleep(5);
X    if (!pid) {
X         setuid(getuid());  /* in case it is setuid-root by mistake */
X         (void)signal(SIGTSTP, SIG_IGN);
X         (void)ioctl(open("/dev/tty", 2), (int)TIOCNOTTY, (char *)0);
X         if ((fd=open(tty, 2)) < 0)
X              _exit(1);
X         (void)dup2(fd, 0);
X         (void)dup2(0, 1);
X         (void)dup2(0, 2);
X         for (fd=3; fd < NFDS; fd++)
X              (void)close(fd);
X         (void)ioctl(0, (int)TIOCSETN, (char *)&sg);
X         (void)ioctl(0, (int)TIOCSETC, (char *)&tc);
X         (void)ioctl(0, (int)TIOCSLTC, (char *)&ltc);
X         (void)ioctl(0, (int)TIOCLSET, (char *)&lmode);
X         if (argc == 1) {
X              if (!(shell=getenv("SHELL")))
X                   shell = "/bin/sh";
X              execl(shell, shell, (char *)0);
X              perror(shell);
X         } else {
X              execvp(argv[1], argv+1);
X              perror(argv[1]);
X         }
X         _exit(1);
X    }
X
X
X    /*
X     * Pass the file descriptor to the window server and exit.
X     */
X
X    iov.iov_base = pty;
X    iov.iov_len = strlen(pty);
X    msg.msg_name = (caddr_t)&sa;
X    msg.msg_namelen = sizeof sa.sa_family + strlen(sa.sa_data);
X    msg.msg_iov = &iov;
X    msg.msg_iovlen = 1;
X    msg.msg_accrights = (caddr_t)&fd;
X    msg.msg_accrightslen = sizeof fd;
X    if (sendmsg(sd, &msg, 0) < 0) {
X         perror("sendmsg");
X         exit(1);
X    }
X
X    exit(0);
X}
```

```
SHAR_EOF
if test 3482 -ne "`wc -c 'uwtool.c'`"
then
     echo shar: error transmitting "'uwtool.c'" '(should have been 3482
characters)'
fi
#    End of shell archive
exit 0

--
usenet: {decvax,cbatt,cbosgd,sun}!cwruecmp!bammi    jwahar r. bammi
csnet:       bammi@case
arpa:        bammi%case@csnet-relay
compuServe:  71515,155


From -7381422231134658021
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,b4f9a898d1703e0b,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1987-08-05 09:45:27 PST
Relay-Version: version B 2.10 5/3/83; site utzoo.UUCP
Path: utzoo!mnetor!uunet!seismo!husc6!rutgers!labrea!decwrl!sun!imagen!atari!
daisy!atari-sources-request
From: atari-sources-request@daisy.UUCP
Newsgroups: comp.sources.atari.st
Subject: v00i024:  mwc:all:uue;arc ZMDM - X/Y/Zmodem xfer & terminal emul.,
Part14/14
Message-ID: <463@daisy.UUCP>
Date: Wed, 5-Aug-87 12:45:27 EDT
Article-I.D.: daisy.463
Posted: Wed Aug  5 12:45:27 1987
Date-Received: Sat, 8-Aug-87 14:00:28 EDT
Sender: turner@daisy.UUCP
Lines: 60
Approved: daisy!turner

Submitted by: sun!cwruecmp!bammi (Jwahar R. Bammi)
comp.sources.atari.st: Volume 0, Issue 24
Archive-name: zmdm/Part14



=+=+=+
```

```
bytes =    19224(    19540)   cksm =  715   zmdd.uue
bytes =    19224(    19540)   cksm =  535   zmde.uue
bytes =    19224(    19540)   cksm = 1696   zmdf.uue
bytes =    19224(    19540)   cksm = 4018   zmdg.uue
bytes =    19224(    19540)   cksm = 2587   zmdh.uue
bytes =    19224(    19540)   cksm = 2146   zmdi.uue
bytes =    19225(    19541)   cksm = 3154   zmdj.uue
bytes =    19225(    19541)   cksm = 3073   zmdk.uue
bytes =    19225(    19541)   cksm =  175   zmdl.uue
bytes =    19225(    19541)   cksm = 1795   zmdm.uue
bytes =     1246(     1278)   cksm = 1822   zmdn.uue

=+=+=+
```

atari-sources-request
From: atari-sources-request@daisy.UUCP
Newsgroups: comp.sources.atari.st
Subject: v00i025: mwc:low: forever.c - linea example & pretty picture gen
Message-ID: <469@daisy.UUCP>
Date: Sun, 9-Aug-87 08:22:16 EDT
Article-I.D.: daisy.469
Posted: Sun Aug 9 08:22:16 1987
Date-Received: Mon, 10-Aug-87 03:49:51 EDT
Sender: turner@daisy.UUCP
Lines: 441
Approved: daisy!turner

Submitted by: <sun!hplabs!tektronix!dadla.TEK.COM!jrb>
comp.sources.atari.st: Volume 0, Issue 25
Archive-name: forever

=+=+=+

```
#define WHITE 0x777
#define BLACK 0x000
#define BLUE  0x007
#define GREEN 0x070
#define RED   0x700

int bluePal[] =
{
0x0777, 0x0124, 0x0214, 0x0324,
0x0424, 0x0417, 0x0336, 0x0517,
0x0437, 0x0716, 0x0727, 0x0700,
0x0704, 0x0705, 0x0706, 0x0707
};

static int gemPal[16];

/* assume 320 by 200
*/
#define CENTERX  160
#define CENTERY    100

/* per "process" pen info
*/

static
struct pp {
     int color;
     int howfat;
} pi[4] = {
     1,0,
     1,0,
     1,0,
     1,0,
};

/* per "process" coords
*/
static
struct chaninfo {
     int cx,cy;
     int opType;
     int xdir,ydir;
} ci[4];

main()
{
     extern int bluePal[];
     int count=0;
     int doit;
     int pal1,pal2;
     int palstate=0;

     openScreen();
```

```
     newPal(bluePal);
     pal1 = 2;
     pal2 = 12;

     /* loop while no keyboard input
     */
     for ( ; Cconis() == 0;count++ ) {
          drunkWalk(0);
          drunkWalk(1);
          drunkWalk(2);
          drunkWalk(3);

          if ( count == 500 )
               if ((rand()%2)==0)
                    doit = 1;
               else {

                    shufflePal(bluePal);
                    doit = 0;
               }

          if ( count > 600 &&
               count < 1000 ){
               if(doit) {
                    if(palstate%2) {
                         pal1 = rand() % 16;
                         pal2 = rand() % 16;
                         twopal(bluePal,pal1,pal2);
                    }
                    else {
                         pal1++;
                         pal2++;
                         if (pal1 > 15)
                              pal1 = 1;
                         if (pal2 > 15)
                              pal2 = 1;
                         twopal(bluePal,pal1,pal2);
                    }
               }
          }

          if ( count > 1000){
               palstate++;
               damage(bluePal);
               count=0;
          }
     }
     closeScreen();
}


/*
 * drunkWalk
```

```
 *
 * implements drunken walk.
 *
 * i is integer index to indicate which mos channel
 * is requesting graphics
 *
 * CALLS:
 *   rand - random number generator.
 *   kplot - plot current drunken walk graphics
 */

static
drunkWalk(i)
{
     register int n,s,t;
     struct chaninfo *cpt;

     cpt = &ci[i];
     cpt->opType = 1;

     /* pick starting delta x,y
     */
     cpt->cx = rand() % 160;
     cpt->cy = rand() % 100;

     n = rand()%127;
     t = rand()%127;
     /* walk change according to note mod 8, mod 12??
     */
     switch(n % 8) {
     case 0:
          cpt->xdir = 1;
          break;
     case 1:
          cpt->xdir = 1;
          cpt->ydir = -1;
          break;
     case 2:
          cpt->xdir = -1;
          break;
     case 3:
          cpt->xdir = -1;
          cpt->ydir = -1;
          break;
     case 4:
          cpt->xdir = 1;
          cpt->ydir = 1;
          break;
     case 5:
          cpt->ydir = 1;
          break;
     case 6:
          cpt->ydir = -1;
```

```
          cpt->ydir = 1;
          break;
     case 7:
          cpt->xdir = -1;
          break;
     }

     kplot(t, i, cpt->cx += cpt->xdir, cpt->cy += cpt->ydir);
     kplot(t, i, cpt->cx += cpt->xdir, cpt->cy += cpt->ydir);
     /* boundary check. If at boundary, we rerandomize
     */
     if ( cpt->cx >= 160 || cpt->cx < 0 )
          cpt->cx = rand() % 30;
     if (cpt->cy >= 100 || cpt->cy < 0 )
               cpt->cy = rand() % 40;
}

/* do graphics work on one pixel. four dots are drawn,
 * one per quadrant.
 */
static
kplot(t, i, deltax,deltay)
int t;       /* event time */
register int i;       /* per coroutine index */
register int deltax,deltay;
{
     register int x,y;
     int chowfat;

     /* randomly switch pen thickness and color
     */
     if ( (rand() % 80) == 20 ) {

          pi[i].howfat = (rand() >> 4) % 4;

          /* randomly choose pen color
          */
          if ( (rand() % 2) == 0 )
               color = t % 16;
          else
               color = rand() % 16;
          if ( color == 0 )
               color++;

     }
     chowfat = pi[i].howfat;

     x = CENTERX+deltax;
     y = CENTERY+deltay;
     setpixel(x,y);
     switch(chowfat) {
     case 0:
          break;
```

```
case 1:
     setpixel(x-1,y);
     setpixel(x+1,y);
     break;

case 2:
     setpixel(x+1,y);
     setpixel(x+1,y+1);
     setpixel(x+1,y-1);
     setpixel(x,y+1);
     setpixel(x,y);
     setpixel(x-1,y);
     setpixel(x,y-1);
     setpixel(x,y-2);
     break;

case 3:
     setpixel(x,y-1);
     setpixel(x,y-2);
     break;
}

x = CENTERX+deltax;
y = CENTERY-deltay;

setpixel(x,y);
switch(chowfat) {
case 0:
     break;

case 1:
     setpixel(x-1,y);
     setpixel(x+1,y);
     break;

case 2:
     setpixel(x+1,y);
     setpixel(x+1,y+1);
     setpixel(x+1,y-1);
     setpixel(x,y+1);
     setpixel(x,y);
     setpixel(x-1,y);
     setpixel(x,y-2);
     setpixel(x,y-1);
     break;
case 3:
     setpixel(x,y-1);
     setpixel(x,y-2);
     break;
}
x = CENTERX-deltax;
y = CENTERY+deltay;
```

```
     setpixel(x,y);
     switch(chowfat) {
     case 0:
          break;

     case 1:
          setpixel(x-1,y);
          setpixel(x+1,y);
          break;

     case 2:
          setpixel(x+1,y);
          setpixel(x+1,y+1);
          setpixel(x+1,y-1);
          setpixel(x,y+1);
          setpixel(x,y);
          setpixel(x-1,y);
          setpixel(x,y-2);
          setpixel(x,y-1);
          break;
     case 3:
          setpixel(x,y-1);
          setpixel(x,y-2);
          break;
     }
     x = CENTERX-deltax;
     y = CENTERY-deltay;
     setpixel(x,y);
     switch(chowfat) {
     case 0:
          break;

     case 1:
          setpixel(x-1,y);
          setpixel(x+1,y);
          break;

     case 2:
          setpixel(x+1,y);
          setpixel(x+1,y+1);
          setpixel(x+1,y-1);
          setpixel(x,y+1);
          setpixel(x,y);
          setpixel(x-1,y);
          setpixel(x,y-2);
          setpixel(x,y-1);
          break;
     case 3:
          setpixel(x,y-1);
          setpixel(x,y-2);
          break;
     }
}
```

```
openScreen()
{
     linea0();
     lineaa();
     Cconws("\033E\033f");    /* erase screen */
}

closeScreen()
{
     Cconin();
     Cconws("\033e\n");
}



/* shift pallete positions. First position remains white.
*/
shufflePal(pal)
int pal[];
{
     int first;
     int index;

     first = pal[1];
     for ( index = 1; index < 15; index++)
          pal[index] = pal[index+1];
     pal[15] = first;
     pal[0] = WHITE;
     newPal(pal);
}

twopal(pal,one,two)
int pal[];
{
     int tmp;
     tmp = pal[one];
     pal[one] = pal[two];
     pal[two] = tmp;
     newPal(pal);
}

/* do something kinky to pal
*/
damage(pal)
int pal[];
{
     register int n;
     register int r,g,b;
     n = rand() % 16;
     r = rand() % 0x7;
     g = rand() % 0x7;
     b = rand() % 0x7;
```

```
     pal[n] = b | (g << 4) |  (r << 8 );
}


/* get default pallete
*/
openPal()
{
     register int i;

     /* get original pallete
     */
     for (i = 0; i < 16; i++)
          gemPal[i] = Setcolor(i, -1);
}

/* restore original pallete
*/
closePal()
{
     Setpallete(gemPal);
}

/* write new pallete
*/
newPal(pal)
int pal[];
{
     Setpallete(pal);
}
```

```
From -154386260010699955
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,bc615b22695844da,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1987-08-17 23:17:18 PST
Relay-Version: version B 2.10 5/3/83; site utzoo.UUCP
Path: utzoo!mnetor!uunet!seismo!lll-lcc!mordor!sri-spam!ames!amdcad!sun!imagen!
atari!daisy!atari-sources-request
From: atari-sources-request@daisy.UUCP
Newsgroups: comp.sources.atari.st
Subject: v00i009:  ::shar - REPOST of UW sources and man page for BSD4.[23]
Message-ID: <379@daisy.UUCP>
Date: Thu, 16-Jul-87 11:45:06 EDT
Article-I.D.: daisy.379
Posted: Thu Jul 16 11:45:06 1987
Date-Received: Sat, 18-Jul-87 11:04:13 EDT
Sender: turner@daisy.UUCP
Lines: 1111
Approved: daisy!turner

Submitted by: ucbvax!cwruecmp!bammi (Jwahar R. Bammi)
```

comp.sources.atari.st: Volume 0, Issue 3
Archive-name: uwvax

+=+=+=

```
X[ process ] [ arg1 arg2 ...  ]
X.SH DESCRIPTION
X.I Uw
Xis a server program on unix that works with the program
X.I uw
Xon the Macintosh.  It creates a pseudo-terminal for each window,
Xgiving the user up to seven `terminals' to work with.  The host
Xprogram multiplexes the output onto one RS\-232 line.
X.PP
X.I Uw
X(on the unix system) with
X.I uw
Xon the Macintosh emulate a Lear Siegler ADM-31 terminal (tset adm31), a
XTektronix 4010, and a DEC VT52.
XEach window (on the Macintosh) has its own terminal emulation and can be
Xresized at will; the program
Xkeeps an 80x24 line display in memory for each window, but only shows the
Xlower left portion possible.
X.PP
X.I Uwtool
Xcan be executed from unix while running
X.I uw
Xto create a new window/`terminal.'  If no arguments are given, it creates
Xa new `terminal' running csh.  If a process is named, it will create a new
Xwindow with that process running in it, and when that process is terminated
Xthe window will disappear.  (i.e. `uwtool vi foo' will create a new window
Xwith vi, editing the file foo, but the window will go away when vi is exited)
XAny arguements after the process name are passed as arguements to that process.
X.SH LIMITATIONS
X.I Uw
Xand
X.I uwtool
Xare of no use on unix unless
X.I uw
Xis being run on the Macintosh.
X.br
XIf there is a stream of output in one window there will be lag in
Xrecognizing characters typed in another.
X.SH SEE ALSO
X.I uw
Xdocumentation distributed with the Macintosh program.
X.SH AUTHOR
XProgram written by John Bruner, Lawrence Livermore Laboratories 7/85
X.br
XDocument edited by Chris Borton, UC San Diego 11/13/85
SHAR_EOF
if test 1732 -ne "`wc -c 'uw.1'`"
then
     echo shar: error transmitting "'uw.1'" '(should have been 1732 characters)'
fi
echo shar: extracting "'uw.c'" '(17040 characters)'
if test -f 'uw.c'
then
```

```
     echo shar: over-writing existing file "'uw.c'"
fi
sed 's/^X//' << \SHAR_EOF > 'uw.c'
X/*
X *  uw - UNIX windows program for the Macintosh (VAX end)
X *
X * Copyright 1985 by John D. Bruner.  All rights reserved.  Permission to
X * copy this program is given provided that the copy is not sold and that
X * this copyright notice is included.
X *
X * Compile: cc -o uw -O uw.c
X */
X
X#include <sys/types.h>
X#include <sys/stat.h>
X#include <sys/socket.h>
X#include <sys/ioctl.h>
X#include <sys/wait.h>
X#include <sys/time.h>
X#include <sys/resource.h>
X#include <sys/uio.h>
X#include <signal.h>
X#include <setjmp.h>
X#include <errno.h>
X#include <fcntl.h>
X#include <stdio.h>
X#include "uw.h"
X
X#define  MAXENV   128           /* maximum environment size */
X
X#define  W_IN 0
X#define  W_OUT    1
X
X#define  NFDS 20            /* max number of file descriptors */
X
X#define  XON  0021          /* ASCII XON */
X#define  XOFF 0023          /* ASCII XOFF */
X#define  RUB  0177          /* ASCII RUBOUT */
X
X#define  META 0200          /* "meta" bit for whatever it's worth */
X
X#define  RCV_OK   (-1)          /* see recvcmd() */
X#define  RCV_META (-2)          /* see recvcmd() */
X
Xtypedef int fildes_t;           /* file descriptor data type */
Xtypedef int nwin_t;             /* window index data type */
X
Xstruct window {
X    fildes_t  w_fd;
X    char      w_tty[32];
X};
X
Xstruct window window[NWINDOW];      /* window data structures */
```

```
Xstruct window *fdmap[NFDS];          /* mapping from fd's to windows */
Xstruct window *curwin[2];       /* current input and output windows */
X
Xchar portal[] = "/tmp/uwXXXXXX";     /* UNIX-domain network address */
X
X
X/* The following are added to the environment of all child processes */
X
Xchar env_uwin[64] = "UWIN=";
Xchar *env[] = {
X    env_uwin,
X    "TERM=adm31",
X    "TERMCAP=adm31:cr=^M:do=^J:nl=^J:al=\\EE:am:le=^H:bs:ce=\\ET:cm=\\E=%+ %
+ :cl=^Z:cd=\\EY:co#80:dc=\\EW:dl=\\ER:ei=\\Er:ho=^^:im=\
\Eq:li#24:mi:nd=^L:up=^K:MT:km:",
X    NULL
X};
X
Xint ctlch[] = { -1, IAC, XON, XOFF, -1, -1, -1, -1 };  /* CTL char mapping */
X
Xstruct selmask {
X    int  sm_rd;
X    int  sm_wt;
X    int  sm_ex;
X} selmask[2];
X
Xextern char *strncpy(), *strncat();
Xextern char *mktemp();
Xextern char *getenv();
Xextern done(), cwait(), onalarm();
Xextern int errno;
X
Xmain(argc, argv)
Xchar **argv;
X{
X    register fildes_t fd, sd;
X    register struct window *w;
X    struct sockaddr sa;
X
X    /*
X     * Make sure we don't accidentally try to run this inside itself.
X     */
X    if (getenv("UWIN")) {
X        fprintf(stderr, "%s is already running\n", *argv);
X        exit(1);
X    }
X
X    /*
X     * Close all file descriptors except 0, 1, and 2.
X     */
X
X    for (fd=3; fd < NFDS; fd++)
X        (void)close(fd);
```

```
X
X    /*
X     * Mark all windows closed.
X     */
X
X    for (w=window; w < window+NWINDOW; w++)
X         w->w_fd = -1;
X
X
X    /*
X     * Create a UNIX-domain network address, and put its name into
X     * the environment so that descendents can contact us with new
X     * window requests.
X     */
X
X    (void)strncat(env_uwin, mktemp(portal), sizeof env_uwin - 1);
X    setenv(env);
X
X    if ((sd=socket(AF_UNIX, SOCK_DGRAM, 0)) < 0) {
X         perror("socket");
X         exit(1);
X    }
X    sa.sa_family = AF_UNIX;
X    (void)strncpy(sa.sa_data, portal, sizeof sa.sa_data-1);
X    sa.sa_data[sizeof sa.sa_data-1] = '\0';
X    if (bind(sd, &sa, sizeof sa.sa_family + strlen(sa.sa_data)) < 0) {
X         perror("bind");
X         exit(1);
X    }
X    if (fcntl(sd, F_SETFL, FNDELAY))
X         perror("fcntl(sd, F_SETFL, FNDELAY)");
X
X
X    /*
X     * Ignore interrupts, quits, and terminal stops.  Clean up and exit
X     * if a hangup or termination is received.  Also catch changes in
X     * child status (so that we can wait for them).  Set up the terminal
X     * modes.
X     */
X
X    (void)signal(SIGHUP, done);
X    (void)signal(SIGINT, SIG_IGN);
X    (void)signal(SIGQUIT, SIG_IGN);
X    (void)signal(SIGTERM, done);
X    (void)signal(SIGTSTP, SIG_IGN);
X    (void)signal(SIGCHLD, cwait);
X
X    tmode(1);
X
X
X    /*
X     * Tell the Macintosh to initialize.  Initialize the current input
X     * and output windows to NULL.
```

```
X     */
X
X    xmitcmd(CB_FN_MAINT|CB_MF_ENTRY);
X    curwin[W_IN] = curwin[W_OUT] = NULL;
X
X
X    /*
X     * Initialize the "select" masks, create window 1 (to start
X     * things off) and wait for input.  When input is available,
X     * process it.
X     */
X
X    selmask[0].sm_rd = (1<<0)|(1<<sd);
X    selmask[0].sm_wt = 0;
X    selmask[0].sm_ex = selmask[0].sm_rd | 2;
X    if (newwind(window) == 0)
X         xmitcmd(CB_FN_NEWW|1);
X
X    while (1) {
X         selmask[1] = selmask[0];
X         if (select(NFDS, &selmask[1].sm_rd, &selmask[1].sm_wt,
X             &selmask[1].sm_ex, (struct timeval *)0) < 0) {
X              if (errno == EINTR)
X                   continue;
X              perror("select");
X              done(1); /* for now -- fix this! */
X         }
X         for (fd=0; fd < NFDS; selmask[1].sm_rd >>= 1, fd++) {
X              if (selmask[1].sm_rd&(1<<0)) {
X                   if (fd < 2)
X                        mrecv();
X                   else if (fd == sd)
X                        netrecv(sd);
X                   else
X                        mxmit(fd);
X              }
X         }
X    }
X}
X
Xmrecv()
X{
X    register int len;
X    register char *cp, *cq;
X    auto int nready;
X    char ibuf[512], obuf[512];
X    static int seen_iac, seen_meta;
X
X    /*
X     * The received bytestream is examined.  Non-command bytes are
X     * written to the file descriptor corresponding to the current
X     * "input" window (relative to the Macintosh -- the window the
X     * user types input to).
```

```
X     */
X
X    if (ioctl(0, (int)FIONREAD, (char *)&nready) < 0) {
X         perror("FIONREAD");
X         return;
X    }
X
X    cq = obuf;
X    while (nready > 0) {
X         if (nready > sizeof ibuf)
X              len = read(0, ibuf, sizeof ibuf);
X         else
X              len = read(0, ibuf, nready);
X         if (len <= 0) {
X              perror("read");
X              return;
X         }
X         for (cp=ibuf; cp < ibuf+len; cp++) {
X              if (seen_iac) {
X                   if ((*cp&CB_DIR) == CB_DIR_MTOH) {
X                        if (cq > obuf) {
X                             (void)write(curwin[W_IN]->w_fd,
X                                      obuf, cq-obuf);
X                             cq = obuf;
X                        }
X                        switch (*cq = recvcmd(*cp)) {
X                        case RCV_OK:
X                             break;
X                        case RCV_META:
X                             seen_meta = 1;
X                             break;
X                        default:
X                             if (seen_meta) {
X                                  seen_meta = 0;
X                                  *cq |= META;
X                             }
X                             if (curwin[W_IN])
X                                  cq++;
X                             break;
X                        }
X                   }
X                   seen_iac = 0;
X              } else if (*cp == IAC)
X                   seen_iac++;
X              else if (curwin[W_IN]) {
X                   if (seen_meta) {
X                        seen_meta = 0;
X                        *cq++ = *cp|META;
X                   } else
X                        *cq++ = *cp;
X                   if (cq >= obuf+sizeof obuf) {
X                        (void)write(curwin[W_IN]->w_fd,
X                                 obuf, cq-obuf);
```

```
X                       cq = obuf;
X                   }
X               }
X          }
X          nready -= len;
X     }
X     if (cq > obuf)
X          (void)write(curwin[W_IN]->w_fd, obuf, cq-obuf);
X}
X
Xrecvcmd(cmd)
Xchar cmd;
X{
X     register int nwin, fn;
X     register struct window *w;
X
X     /*
X      * Perform the function the Mac is asking for.  There are three
X      * broad categories of these functions: RCV_META, which tells
X      * the caller that the next character is a "meta" character;
X      * an ASCII data character, which is passed back to the caller
X      * for proper handling; and RCV_OK, which means that this routine
X      * has done everything which was required to process the command.
X      */
X
X     fn = cmd&CB_FN;
X     switch (fn) {
X     case CB_FN_NEWW:
X     case CB_FN_KILLW:
X     case CB_FN_ISELW:
X          nwin = cmd&CB_WINDOW;
X          if (!nwin)
X               break;
X          w = &window[nwin-1];
X
X          switch (fn) {
X          case CB_FN_NEWW:
X               if (w->w_fd < 0 && newwind(w) < 0)
X                    xmitcmd(CB_FN_KILLW|nwin);
X               break;
X
X          case CB_FN_KILLW:
X               killwind(w, 0);
X               break;
X
X          case CB_FN_ISELW:
X               if (w->w_fd >= 0)
X                    curwin[W_IN] = w;
X               break;
X          }
X          break;
X
X     case CB_FN_META:
```

```
X         return(RCV_META);
X
X    case CB_FN_CTLCH:
X         return(ctlch[cmd&CB_CC]);
X
X    case CB_FN_MAINT:
X         if ((cmd&CB_MF) == CB_MF_EXIT)
X              done(0);
X         /*NOTREACHED*/
X    }
X    return(RCV_OK);
X}
X
Xxmitcmd(cmd)
Xchar cmd;
X{
X    static char cmdbuf[2] = { IAC, '\0' };
X
X    /*
X     * Transmit the command "cmd" to the Macintosh.  The byte is ORed
X     * with the host-to-Mac direction indicator.
X     */
X
X    cmdbuf[1] = cmd|CB_DIR_HTOM;
X    (void)write(1, cmdbuf, sizeof cmdbuf);
X}
X
Xnetrecv(sd)
Xregister fildes_t sd;
X{
X    register struct window *w;
X    register int cnt;
X    struct msghdr msg;
X    auto int fd;
X    struct iovec iov;
X    struct stat st1, st2;
X    static int unity = 1;
X    char buf[256];
X
X
X    /*
X     * main() calls this routine when there is a message waiting on
X     * the UNIX-domain socket.  The message's access rights are
X     * expected to contain the file descriptor for the "master" side
X     * of a pseudo-tty.  The message contains the name of the pty.
X     * The sender is expected to start up a process on the slave side
X     * of the pty.  This allows the host end to create windows which
X     * run something other than the shell.
X     */
X
X    fd = -1;
X
X    iov.iov_base = (caddr_t)buf;
```

```
X    iov.iov_len = sizeof buf - 1;
X
X    msg.msg_name = (caddr_t)0;
X    msg.msg_namelen = 0;
X    msg.msg_iov = &iov;
X    msg.msg_iovlen = 1;
X    msg.msg_accrights = (caddr_t)&fd;
X    msg.msg_accrightslen = sizeof fd;
X
X    if ((cnt=recvmsg(sd, &msg, 0)) < 0)
X         perror("recvmsg");
X
X    if (msg.msg_accrightslen > 0 && fd >= 0) {
X         /*
X          * We can't trust the process which connected to us, so
X          * we verify that it really passed us a pseudo-tty's
X          * file descriptor by checking the device name and its
X          * inode number.  [Of course, if someone else wants to
X          * hand us a terminal session running under their uid....]
X          */
X         buf[cnt] = 0;
X         if (strncmp(buf, "/dev/ptyp", sizeof "/dev/ptyp" - 1) ||
X             fstat(fd, &st1) < 0 || stat(buf, &st2) < 0 ||
X             st1.st_dev != st2.st_dev || st1.st_ino != st2.st_ino) {
X              (void)close(fd);
X              return;
X         }
X         /*
X          * OK, we believe the sender.  We allocate a window and
X          * tell the Macintosh to create that window on its end.
X          */
X         buf[5] = 't';      /* switch to "/dev/ttyp?" */
X         for (w=window; w < window+NWINDOW; w++) {
X              if (w->w_fd < 0) {
X                   w->w_fd = fd;
X                   fdmap[fd] = w;
X                   selmask[0].sm_rd |= (1<<fd);
X                   (void)strncpy(w->w_tty, buf, sizeof w->w_tty-1);
X                   xmitcmd(CB_FN_NEWW|(w-window+1));
X                   break;
X              }
X         }
X         /*
X          * If we have no free windows, then we close the file
X          * descriptor (which will terminate the slave process).
X          */
X         if (w == window+NWINDOW)
X              (void)close(fd);
X         /*
X          * Set non-blocking I/O on the master side.
X          */
X         (void)ioctl(fd, FIONBIO, &unity);
X    }
```

```
X}
X
Xmxmit(fd)
Xregister fildes_t fd;
X{
X    register char *cp, *cq, i;
X    register int len;
X    char ibuf[16], obuf[16];
X
X    /*
X     * Copy input from file "fd" to the Macintosh.  Be sure to convert
X     * any embedded IAC characters.
X     *
X     * Note that the input/output buffers should NOT be very large.
X     * It is undesirable to perform large reads and effectively
X     * "lock out" all other file descriptors.  The chosen size
X     * should preserve a reasonable amount of efficiency.
X     */
X
X    if (fdmap[fd]) {
X         curwin[W_OUT] = fdmap[fd];
X         xmitcmd(CB_FN_OSELW|(fdmap[fd]-window+1));
X         cq = obuf;
X         if ((len = read(fd, ibuf, sizeof ibuf)) < 0 &&
X             errno != EWOULDBLOCK) {
X              killwind(fdmap[fd], 1);
X              return;
X         }
X         for (cp=ibuf; cp < ibuf+len; cp++) {
X              if (*cp&META) {
X                   if (cq > obuf) {
X                        (void)write(1, obuf, cq-obuf);
X                        cq = obuf;
X                   }
X                   xmitcmd(CB_FN_META);
X                   *cp &= ~META;
X              }
X              i = -1;
X              if (*cp == RUB || *cp < ' ') {
X                   i = sizeof ctlch;
X                   while (i >= 0 && ctlch[i] != *cp)
X                        i--;
X              }
X              if (i >= 0) {
X                   if (cq > obuf) {
X                        (void)write(1, obuf, cq-obuf);
X                        cq = obuf;
X                   }
X                   xmitcmd(CB_FN_CTLCH|i);
X              } else {
X                   *cq++ = *cp;
X                   if (cq >= obuf+sizeof obuf) {
X                        (void)write(1, obuf, cq-obuf);
```

```
X                       cq = obuf;
X                   }
X               }
X           }
X       } else
X           (void)read(fd, ibuf, sizeof ibuf);
X       if (cq > obuf)
X           (void)write(1, obuf, cq-obuf);
X}
X
Xkillwind(w, notify)
Xregister struct window *w;
X{
X   register int mask;
X
X   /*
X    * Kill window "w".  Notify the Macintosh that it is gone if
X    * "notify" is nonzero.
X    */
X
X   (void)close(w->w_fd);
X   mask = ~(1<<w->w_fd);
X   fdmap[w->w_fd] = NULL;
X   w->w_fd = -1;
X   if (curwin[W_IN] == w)
X       curwin[W_IN] = NULL;
X   if (curwin[W_OUT] == w)
X       curwin[W_OUT] = NULL;
X   selmask[0].sm_rd &= mask;
X   selmask[0].sm_wt &= mask;
X   selmask[0].sm_ex &= mask;
X   if (notify)
X       xmitcmd(CB_FN_KILLW|(w-window+1));
X}
X
Xnewwind(w)
Xregister struct window *w;
X{
X   register char *cp;
X   register fildes_t fd;
X   register int pid;
X   register char *shell;
X   char pty[32], ptysufx[2];
X   static char ptyidx[] = "0123456789abcdefghijklmnopqrstuvwxyz";
X   static int unity = 1;
X   extern char *getenv();
X
X   /*
X    * Create a new window using the specified component of "window".
X    * This routine isn't very smart at finding pseudo-ttys.
X    */
X
X   ptysufx[1] = '\0';
```

```
X    for (cp=ptyidx; *cp; cp++) {
X         ptysufx[0] = *cp;
X         (void)strncpy(pty, "/dev/ptyp", sizeof pty-1);
X         (void)strncat(pty, ptysufx, sizeof pty-1);
X         if ((fd = open(pty, 2)) >= 0)
X              break;
X    }
X    if (fd < 0)
X         return(-1);
X    (void)ioctl(fd, FIONBIO, &unity);     /* set non-blocking I/O */
X    fdmap[fd] = w;
X    w->w_fd = fd;
X    selmask[0].sm_rd |= (1<<fd);
X    (void)strncpy(w->w_tty, "/dev/ttyp", sizeof w->w_tty-1);
X    (void)strncat(w->w_tty, ptysufx, sizeof w->w_tty-1);
X
X    while ((pid=fork()) < 0)
X         sleep(5);
X    if (!pid) {
X         (void)signal(SIGHUP, SIG_DFL);
X         (void)signal(SIGINT, SIG_DFL);
X         (void)signal(SIGQUIT, SIG_DFL);
X         (void)signal(SIGTERM, SIG_DFL);
X         (void)signal(SIGTSTP, SIG_IGN);
X         (void)signal(SIGCHLD, SIG_DFL);
X         (void)ioctl(open("/dev/tty", 2), (int)TIOCNOTTY, (char *)0);
X         (void)setuid(getuid());     /* shouldn't need this */
X         if ((fd=open(w->w_tty, 2)) < 0)
X              _exit(1);
X         if (!(shell = getenv("SHELL")))
X              shell = "/bin/sh";
X         (void)dup2(fd, 0);
X         (void)dup2(0, 1);
X         (void)dup2(0, 2);
X         for (fd=3; fd < NFDS; fd++)
X              (void)close(fd);
X         tmode(0);/* HACK! */
X         execl(shell, shell, (char *)0);
X         _exit(1);
X    }
X
X    return(0);
X}
X
Xtmode(f)
X{
X    static struct sgttyb ostty, nstty;
X    static struct tchars otchars, ntchars;
X    static struct ltchars oltchars, nltchars;
X    static int olmode, nlmode;
X    static saved;
X
X    /*
```

```
X     * This routine either saves the current terminal modes and then
X     * sets up the terminal line or resets the terminal modes (depending
X     * upon the value of "f").  The terminal line is used in "cbreak"
X     * mode with all special characters except XON/XOFF disabled.  The
X     * hated (by me) LDECCTQ mode is required for the Macintosh to
X     * handle flow control properly.
X     */
X
X    if (f == 1) {
X         if (ioctl(0, (int)TIOCGETP, (char *)&ostty) < 0) {
X              perror("ioctl((int)TIOCGETP)");
X              done(1);
X         }
X         if (ioctl(0, (int)TIOCGETC, (char *)&otchars) < 0) {
X              perror("ioctl((int)TIOCGETC)");
X              done(1);
X         }
X         if (ioctl(0, (int)TIOCGLTC, (char *)&oltchars) < 0) {
X              perror("ioctl((int)TIOCGLTC)");
X              done(1);
X         }
X         if (ioctl(0, (int)TIOCLGET, (char *)&olmode) < 0) {
X              perror("ioctl((int)TIOCLGET)");
X              done(1);
X         }
X         nstty = ostty;
X         nstty.sg_erase = nstty.sg_kill = -1;
X         nstty.sg_flags |= CBREAK;
X         nstty.sg_flags &= ~(RAW|CRMOD|ECHO|LCASE|XTABS);
X         ntchars.t_intrc = ntchars.t_quitc = -1;
X         ntchars.t_eofc = ntchars.t_brkc = -1;
X         ntchars.t_startc = XON;
X         ntchars.t_stopc = XOFF;
X         nltchars.t_suspc = nltchars.t_dsuspc = -1;
X         nltchars.t_rprntc = nltchars.t_flushc = -1;
X         nltchars.t_werasc = nltchars.t_lnextc = -1;
X         nlmode = olmode | LDECCTQ;
X         if (ioctl(0, (int)TIOCSETN, (char *)&nstty) < 0) {
X              perror("ioctl((int)TIOCSETN)");
X              done(1);
X         }
X         if (ioctl(0, (int)TIOCSETC, (char *)&ntchars) < 0) {
X              perror("ioctl((int)TIOCSETC");
X              done(1);
X         }
X         if (ioctl(0, (int)TIOCSLTC, (char *)&nltchars) < 0) {
X              perror("ioctl((int)TIOCSLTC");
X              done(1);
X         }
X         if (ioctl(0, (int)TIOCLSET, (char *)&nlmode) < 0) {
X              perror("ioctl((int)TIOCLSET)");
X              done(1);
X         }
```

```
X        saved = 1;
X    } else if (saved) {
X        (void)ioctl(0, (int)TIOCSETP, (char *)&ostty);
X        (void)ioctl(0, (int)TIOCSETC, (char *)&otchars);
X        (void)ioctl(0, (int)TIOCSLTC, (char *)&oltchars);
X        (void)ioctl(0, (int)TIOCLSET, (char *)&olmode);
X    }
X}
X
Xdone(s)
X{
X    register struct window *w;
X    register fildes_t fd;
X
X    /*
X     * Clean up and exit.  It is overkill to close all of the file
X     * descriptors, but it causes no harm.  After we are sure that
X     * our UNIX-domain network connection is closed we remove the
X     * entry that it created (as a side effect) in the filesystem.
X     *
X     * We also restore the terminal modes.
X     */
X
X    /*xmitcmd(CB_FN_MAINT|CB_MF_EXIT);*/
X    for (fd=3; fd < NFDS; fd++)
X        (void)close(fd);
X    (void)unlink(portal);
X    tmode(0);
X    exit(s);
X}
X
Xcwait()
X{
X    union wait status;
X    struct rusage rusage;
X
X    /*
X     * Collect dead children.  We don't use the information that
X     * wait3() returns.  (Someday we might.)
X     */
X
X    while (wait3(&status, WNOHANG, &rusage) > 0)
X        ;
X}
X
X
Xstatic char *earray[MAXENV+1];
X
Xsetenv(env)
Xchar **env;
X{
X    register char **ep1, **ep2, *cp;
X    char **ep3;
```

```
X    extern char **environ;
X
X
X    /*
X     * Merge the set of environment strings in "env" into the
X     * environment.
X     */
X
X    /*
X     * The first time through, copy the environment from its
X     * original location to the array "earray".  This makes it a
X     * little easier to change things.
X     */
X
X    if (environ != earray){
X         ep1=environ;
X         ep2=earray;
X         while(*ep1 && ep2 <= earray+MAXENV)
X              *ep2++ = *ep1++;
X         *ep2++ = NULL;
X         environ = earray;
X    }
X
X
X    /*
X     * If "env" is non-NULL, it points to a list of new items to
X     * be added to the environment.  These replace existing items
X     * with the same name.
X     */
X
X    if (env){
X         for(ep1=env; *ep1; ep1++){
X              for(ep2=environ; *ep2; ep2++)
X                   if (!envcmp(*ep1, *ep2))
X                        break;
X              if (ep2 < earray+MAXENV) {
X                   if (!*ep2)
X                        ep2[1] = NULL;
X                   *ep2 = *ep1;
X              }
X         }
X    }
X
X
X    /* Finally, use an insertion sort to put things in order. */
X
X    for(ep1=environ+1; cp = *ep1; ep1++){
X         for(ep2=environ; ep2 < ep1; ep2++)
X              if (envcmp(*ep1, *ep2) < 0)
X                   break;
X         ep3 = ep2;
X         for(ep2=ep1; ep2 > ep3; ep2--)
X              ep2[0] = ep2[-1];
```

```
X         *ep2 = cp;
X    }
X}
X
X
Xstatic
Xenvcmp(e1, e2)
Xregister char *e1, *e2;
X{
X    register d;
X
X    do {
X         if (d = *e1 - *e2++)
X              return(d);
X    } while(*e1 && *e1++ != '=');
X    return(0);
X}
SHAR_EOF
if test 17040 -ne "`wc -c 'uw.c'`"
then
     echo shar: error transmitting "'uw.c'" '(should have been 17040 characters)'
fi
echo shar: extracting "'uw.h'" '(2697 characters)'
if test -f 'uw.h'
then
     echo shar: over-writing existing file "'uw.h'"
fi
sed 's/^X//' << \SHAR_EOF > 'uw.h'
X/*
X *  uw command bytes
X *
X * Copyright 1985 by John D. Bruner.  All rights reserved.  Permission to
X * copy this program is given provided that the copy is not sold and that
X * this copyright notice is included.
X *
X *
X * Two types of information are exchanged through the 7-bit serial line:
X * ordinary data and command bytes.  Command bytes are preceeded by
X * an IAC byte.  IAC bytes and literal XON/XOFF characters (those which
X * are not used for flow control) are sent by a CB_FN_CTLCH command.
X * Characters with the eighth bit set (the "meta" bit) are prefixed with
X * a CB_FN_META function.
X *
X * The next most-significant bit in the byte specifies the sender and
X * recipient of the command.  If this bit is clear (0), the command byte
X * was sent from the host computer to the Macintosh; if it is set (1)
X * the command byte was sent from the Macintosh to the host computer.
X * This prevents confusion in the event that the host computer
X * (incorrectly) echos a command back to the Macintosh.
X *
X * The remaining six bits are partitioned into two fields.  The low-order
X * three bits specify a window number from 1-7 (window 0 is reserved for
X * other uses) or another type of command-dependent parameter.  The next
```

```
X * three bits specify the operation to be performed by the recipient of
X * the command byte.
X *
X * Note that the choice of command bytes prevents the ASCII XON (021) and
X * XOFF (023) characters from being sent as commands.  CB_FN_ISELW commands
X * are only sent by the Macintosh (and thus are tagged with the CB_DIR_MTOH
X * bit).  Since XON and XOFF data characters are handled via CB_FN_CTLCH,
X * this allows them to be used for flow control purposes.
X */
X
X#define  IAC       0001     /* interpret as command */
X#define  CB_DIR         0100     /* command direction: */
X#define  CB_DIR_HTOM    0000     /*   from host to Mac */
X#define  CB_DIR_MTOH    0100     /*   from Mac to host */
X#define  CB_FN          0070     /* function code: */
X#define  CB_FN_NEWW     0000     /*   new window */
X#define  CB_FN_KILLW    0010     /*   kill (delete) window */
X#define  CB_FN_ISELW    0020     /*   select window for input */
X#define  CB_FN_OSELW    0030     /*   select window for output */
X#define  CB_FN_META     0050     /*   add meta to next data char */
X#define  CB_FN_CTLCH    0060     /*   low 3 bits specify char */
X#define  CB_FN_MAINT    0070     /*   maintenance functions */
X#define  CB_WINDOW 0007     /* window number mask */
X#define  CB_CC          0007     /* control character specifier: */
X#define  CB_CC_IAC 1         /*   IAC */
X#define  CB_CC_XON 2         /*   XON */
X#define  CB_CC_XOFF     3         /*   XOFF */
X#define  CB_MF          0007     /* maintenance functions: */
X#define  CB_MF_ENTRY    0         /*   beginning execution */
X#define  CB_MF_EXIT     7         /*   execution terminating */
X#define  NWINDOW        7         /* maximum number of windows */
SHAR_EOF
if test 2697 -ne "`wc -c 'uw.h'`"
then
     echo shar: error transmitting "'uw.h'" '(should have been 2697 characters)'
fi
echo shar: extracting "'uwtool.c'" '(3482 characters)'
if test -f 'uwtool.c'
then
     echo shar: over-writing existing file "'uwtool.c'"
fi
sed 's/^X//' << \SHAR_EOF > 'uwtool.c'
X#
X
X/*
X *  uwtool
X *
X * Copyright 1985 by John D. Bruner.  All rights reserved.  Permission to
X * copy this program is given provided that the copy is not sold and that
X * this copyright notice is included.
X *
X * Compile: cc -o uwtool -O uwtool.c
X */
```

```
X
X#include <sys/types.h>
X#include <sys/socket.h>
X#include <sys/ioctl.h>
X#include <sys/wait.h>
X#include <sys/time.h>
X#include <sys/resource.h>
X#include <sys/uio.h>
X#include <signal.h>
X#include <stdio.h>
X
X#define  NFDS 20            /* max number of file descriptors */
X
Xtypedef int fildes_t;
X
Xextern char *strncpy(), *strncat();
Xextern char *getenv();
X
Xmain(argc, argv)
Xchar **argv;
X{
X    register char *cp;
X    register int pid;
X    register fildes_t sd;
X    auto fildes_t fd;
X    char *portal, *shell;
X    char tty[32], pty[32], ptysufx[2];
X    int lmode;
X    struct sgttyb sg;
X    struct tchars tc;
X    struct ltchars ltc;
X    struct iovec iov;
X    struct msghdr msg;
X    static char ptyidx[] = "0123456789abcdefghijklmnopqrstuvwxyz";
X    struct sockaddr sa;
X
X    /*
X     * Close all file descriptors except 0, 1, and 2.
X     */
X
X    for (fd=3; fd < NFDS; fd++)
X        (void)close(fd);
X
X    /*
X     * Get the terminal configuration for this tty.
X     */
X    (void)ioctl(0, (int)TIOCGETP, (char *)&sg);
X    (void)ioctl(0, (int)TIOCGETC, (char *)&tc);
X    (void)ioctl(0, (int)TIOCGLTC, (char *)&ltc);
X    (void)ioctl(0, (int)TIOCLGET, (char *)&lmode);
X
X    /*
X     * Create a UNIX-domain socket.
```

```
X     */
X
X    if (!(portal=getenv("UWIN"))) {
X         fprintf(stderr,
X             "You must run %s under the window manager\n", *argv);
X         exit(1);
X    }
X
X    if ((sd=socket(AF_UNIX, SOCK_DGRAM, 0)) < 0) {
X         perror("socket");
X         exit(1);
X    }
X    sa.sa_family = AF_UNIX;
X    (void)strncpy(sa.sa_data, portal, sizeof sa.sa_data-1);
X    sa.sa_data[sizeof sa.sa_data-1] = '\0';
X
X
X    /*
X     * Obtain a pseudo-tty.  This code isn't very smart about finding one.
X     */
X
X    ptysufx[1] = '\0';
X    for (cp=ptyidx; *cp; cp++) {
X         ptysufx[0] = *cp;
X         (void)strncpy(pty, "/dev/ptyp", sizeof pty-1);
X         (void)strncat(pty, ptysufx, sizeof pty-1);
X         if ((fd = open(pty, 2)) >= 0)
X              break;
X    }
X    if (fd < 0) {
X         fprintf(stderr, "Can't obtain a pseudo-tty for a new window\n");
X         exit(1);
X    }
X    (void)strncpy(tty, "/dev/ttyp", sizeof tty-1);
X    (void)strncat(tty, ptysufx, sizeof tty-1);
X
X
X    /*
X     * Fork a child process using this pseudo-tty.  Initialize the
X     * terminal modes on the pseudo-tty to match those of the parent
X     * tty.
X     */
X
X    while ((pid=fork()) < 0)
X         sleep(5);
X    if (!pid) {
X         setuid(getuid());  /* in case it is setuid-root by mistake */
X         (void)signal(SIGTSTP, SIG_IGN);
X         (void)ioctl(open("/dev/tty", 2), (int)TIOCNOTTY, (char *)0);
X         if ((fd=open(tty, 2)) < 0)
X              _exit(1);
X         (void)dup2(fd, 0);
X         (void)dup2(0, 1);
```

```
X        (void)dup2(0, 2);
X        for (fd=3; fd < NFDS; fd++)
X             (void)close(fd);
X        (void)ioctl(0, (int)TIOCSETN, (char *)&sg);
X        (void)ioctl(0, (int)TIOCSETC, (char *)&tc);
X        (void)ioctl(0, (int)TIOCSLTC, (char *)&ltc);
X        (void)ioctl(0, (int)TIOCLSET, (char *)&lmode);
X        if (argc == 1) {
X             if (!(shell=getenv("SHELL")))
X                  shell = "/bin/sh";
X             execl(shell, shell, (char *)0);
X             perror(shell);
X        } else {
X             execvp(argv[1], argv+1);
X             perror(argv[1]);
X        }
X        _exit(1);
X    }
X
X
X    /*
X     * Pass the file descriptor to the window server and exit.
X     */
X
X    iov.iov_base = pty;
X    iov.iov_len = strlen(pty);
X    msg.msg_name = (caddr_t)&sa;
X    msg.msg_namelen = sizeof sa.sa_family + strlen(sa.sa_data);
X    msg.msg_iov = &iov;
X    msg.msg_iovlen = 1;
X    msg.msg_accrights = (caddr_t)&fd;
X    msg.msg_accrightslen = sizeof fd;
X    if (sendmsg(sd, &msg, 0) < 0) {
X         perror("sendmsg");
X         exit(1);
X    }
X
X    exit(0);
X}
SHAR_EOF
if test 3482 -ne "`wc -c 'uwtool.c'`"
then
     echo shar: error transmitting "'uwtool.c'" '(should have been 3482
characters)'
fi
#    End of shell archive
exit 0

--
usenet: {decvax,cbatt,cbosgd,sun}!cwruecmp!bammi    jwahar r. bammi
csnet:       bammi@case
arpa:        bammi%case@csnet-relay
compuServe:  71515,155
```

From 7505610491886320299
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,66e19e7d79357b1d,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1987-11-08 10:53:43 PST
Relay-Version: version B 2.10 5/3/83; site utzoo.UUCP
Path: utzoo!mnetor!uunet!lll-winken!lll-lcc!ames!sgi!daisy!atari-sources-request
From: atari-sources-request@daisy.UUCP
Newsgroups: comp.sources.atari.st
Subject: v00i034: :: TERMCAP - termcap file for temcap emulation
Message-ID: <592@daisy.UUCP>
Date: Sun, 8-Nov-87 13:53:43 EST
Article-I.D.: daisy.592
Posted: Sun Nov 8 13:53:43 1987
Date-Received: Tue, 10-Nov-87 06:48:18 EST
Sender: turner@daisy.UUCP
Lines: 24
Approved: daisy!turner

comp.sources.atari.st: Volume 0, Issue 34
Archive-name: termcap.1

+=+=+=

Relay-Version: version B 2.10 5/3/83; site utzoo.UUCP
Path: utzoo!mnetor!uunet!husc6!cmcl2!nrl-cmf!ames!sgi!daisy!atari-sources-request
From: atari-sources-request@daisy.UUCP
Newsgroups: comp.sources.atari.st
Subject: v00i035: bas:all: UUCODE.BAS - basic version of uudecode
Message-ID: <593@daisy.UUCP>
Date: Mon, 9-Nov-87 05:17:29 EST
Article-I.D.: daisy.593
Posted: Mon Nov 9 05:17:29 1987
Date-Received: Wed, 11-Nov-87 05:52:15 EST
Sender: turner@daisy.UUCP
Lines: 111
Approved: daisy!turner

Submitted by: imagen!ucbvax!decvax!sbcs!lean (Lean L. Loh)
comp.sources.atari.st: Volume 0, Issue 35
Archive-name: uucode.bas

=++=++=++

```
40    INPUT "Enter name of uuencoded file > ", IFILEN$
50    OPEN "I", #1, IFILEN$

60    ' Get output filename, open, and set up pointers.
70    INPUT "Enter name of output file > ", OFILEN$
80    OPEN "R", #2, OFILEN$, 45
90    FIELD #2, 45 AS CBUF$

110   ' Input and ignore first line of input file (original filename).
120   LINE INPUT #1, INLINE$ ' Due to a bug in ST BASIC, LINE INPUT #
130   LINE INPUT #1, INLINE$ '  must be called 2x, just to get first
140                          '  line. Even then we lose the 1st character.

150   ' Now set up screen for user.
160   CLEARW 2
170   GOTOXY 1,2 : PRINT "Decoding "; IFILEN$; " to "; OFILEN$
180   GOTOXY 1,4 : PRINT "# lines read in: "
190   GOTOXY 1,5 : PRINT "# bytes output: "

200   ' Initialize some variables for main body of program.
210   NUML = 0 : BYTEOUT = 0

220   LINELOOP: ' Now loop for as long as there are input lines.
230      LINE INPUT #1, INLINE$           ' LINE INPUT should work OK by now!
231      IF INLINE$ = "end" THEN GOTO 500 ' All done
232      NUML = NUML + 1

240   ' For each line, must "decode" each byte.
241   ' First byte contains # bytes to ***output*** from this line.
242   ' Then, after decoding, each set of 4 input bytes contains 6*4 = 24
243   '  bits of information to output, 8 bits per output byte, thus
244   '  3 output bytes.

250   ' Set up string position pointer for INLINE$
260      SPTR = 1

270   ' Get first byte = # bytes to output; exit if line has zero to output
280      GOSUB DECODE : NN = C : N = NN : IF NN = 0 THEN GOTO 500

285   ' Process one line of input
290      MPTR = 1
300      WHILE N > 0          ' Decode set of 4 input bytes.
310         FOR J = 1 TO 4
320            GOSUB DECODE : I(J) = C
330         NEXT J
340         C1 = ((I(1) *  4) MOD 256) + INT(I(2) / 16)
350         C2 = ((I(2) * 16) MOD 256) + INT(I(3) /  4)
360         C3 = ((I(3) * 64) MOD 256) + I(4)
370         IF N>0 THEN MID$(CBUF$,MPTR,1)=CHR$(C1) : MPTR = MPTR + 1
380         IF N>1 THEN MID$(CBUF$,MPTR,1)=CHR$(C2) : MPTR = MPTR + 1
390         IF N>2 THEN MID$(CBUF$,MPTR,1)=CHR$(C3) : MPTR = MPTR + 1
400         N = N - 3
410      WEND                 ' If more in this line, go back again.
```

```
415   ' Finished processing line.
420      PUT #2, NUML               ' Send decoded stuff to output file.
430      BYTEOUT = BYTEOUT + NN     ' Show status to user.
440      GOTOXY 20,4 : PRINT NUML
445      GOTOXY 20,5 : PRINT BYTEOUT
450   GOTO LINELOOP                 ' Get next line ...

500   ' Hopefully, we have now decoded the input file.
510   CLOSE #1 : CLOSE #2
520   END

1000  DECODE: ' Subroutine to decode a single character
1010     IF LEN(MID$(INLINE$,SPTR)) = 0 THEN INLINE$ = INLINE$ + " "
1020     CH$ = MID$(INLINE$, SPTR, 1) : SPTR = SPTR + 1
1030     C = (ASC(CH$) - SP) AND MASK
1040  RETURN
```

==========

```
From 955700831167061117
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,43d061e532e6b5e3,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1987-11-28 19:39:10 PST
Relay-Version: version B 2.10 5/3/83; site utzoo.UUCP
Path: utzoo!mnetor!uunet!seismo!sundc!pitstop!sun!decwrl!sgi!daisy!atari-sources-
request
From: atari-sources-request@daisy.UUCP
Newsgroups: comp.sources.atari.st
Subject: v00i047:  mmc:all: IBMFMT.C - program to format IBM disks usable by ST
Message-ID: <674@daisy.UUCP>
Date: Sat, 28-Nov-87 22:39:10 EST
Article-I.D.: daisy.674
Posted: Sat Nov 28 22:39:10 1987
Date-Received: Tue, 1-Dec-87 06:35:30 EST
Sender: turner@daisy.UUCP
Lines: 218
Approved: daisy!turner

Submitted by: imagen!decwrl!hplabs!ames!rochester!cornell!batcomputer!braner
(braner)
comp.sources.atari.st: Volume 0, Issue 47
Archive-name: ibmfmt.c



~-~-~-
```

uses 3.5" disks nor do I have a 5 1/4" B: disk. The program does compile
under MegaMax C and runs as far as I can test it.

the checksum is:

```
bytes =     5891(     6071)   cksm = 1030   ibmfmtr.c
```

~~~~~~

Hi, James!

Here is a submission for comp.*.atari.st (some for each):

This is a corrected version of the 200% MS-DOS-compatible formatter.
It will format 3.5" (720K, MS-DOS 3.3) disks in drive A and 5.25" (360K,
MS-DOS 2.0) disks in drive B (assuming you have a 5.25" drive attached
as drive B!). (Sorry, no frills!) The disks are NOT bootable (on MS-DOS)
but both the Atari ST and MS-DOS machines can read and write files on
them, even in subdirectories. The correction (of the early version that
I posted to comp.sys.atari.st) is that now a random disk ID is created.
In the early version, the same ID was always used (stupid me!).

Thanks for doing the moderation job (must be a pain)!

- Moshe Braner

Here is the source code (Megamax C), followed by the uuencoded binary.

```
------------------ cut here ------------------
/**
 **  Program to format MS-DOS-compatible disks on the Atari ST
 **  Code for Megamax C compiler 871023
 **/

#include <osbind.h>

#define WORD int       /* 16 bits!!! */

/*
 *  IBM style boot sector for nonsystem 5.25" disk:
 *
 *  (This is simply:
 *   WORD boot5[256] = { 0xEB3C, ... };
 *   done in a sick way to bypass the sick element-by-element
 *   array-initialization scheme used by Megamax...)
 */
extern boot5();
asm{
boot5:
DC.W 0xEB3C,0x9049,0x424D,0x2020,0x322E,0x3000,0x0202,0x0100
DC.W 0x0270,0x00D0,0x02FD,0x0200,0x0900,0x0200,0x0000,0x2020
DC.W 0x2020,0x2020,0x2020,0x2020,0x2020,0x2020,0x2020,0x2020
DC.W 0x2020,0x2020,0x2020,0x2020,0x2020,0x2020,0x2020,0x0E1F
DC.W 0x33C0,0xFA8E,0xD0BC,0x007C,0x36C7,0x0678,0x00F4,0x7C36
```
~~~~~~

```
DC.W 0xA37A,0x00FB,0xCD13,0x7264,0xB850,0x008E,0xC033,0xDBB8
DC.W 0x0102,0xB906,0x0033,0xD2CD,0x1372,0x51B1,0x0B26,0x8A07
DC.W 0x3A87,0xDE7C,0x754E,0x268A,0x4720,0x3A87,0xE97C,0x7544
DC.W 0x43E2,0xEAB8,0x7000,0x8EC0,0x33DB,0xB806,0x02B1,0x04B6
DC.W 0x01CD,0x1372,0x27B7,0x0CB8,0x0902,0xB901,0x01B6,0x00CD
DC.W 0x1372,0x19CD,0x11D0,0xC0D0,0xC025,0x0300,0x7501,0x4040
DC.W 0x8BC8,0x33C0,0xBB0C,0x00EA,0x0000,0x7000,0xBEFF,0x7CE8
DC.W 0x0E00,0xEBFE,0xBE15,0x7DE8,0x0600,0xB400,0xCD16,0xCD19
DC.W 0xAC84,0xC074,0x08B4,0x0EB3,0x07CD,0x10EB,0xF3C3,0x4942
DC.W 0x4D42,0x494F,0x2020,0x434F,0x4D49,0x424D,0x444F,0x5320
DC.W 0x2043,0x4F4D,0xDF02,0x2502,0x092A,0xFF50,0x000F,0x020D
DC.W 0x0A44,0x6973,0x6B20,0x426F,0x6F74,0x2066,0x6169,0x6C75
DC.W 0x7265,0x0D0A,0x000D,0x0A4E,0x6F6E,0x2D53,0x7973,0x7465
DC.W 0x6D20,0x6469,0x736B,0x206F,0x7220,0x6469,0x736B,0x2065
DC.W 0x7272,0x6F72,0x0D0A,0x0D0A,0x5265,0x706C,0x6163,0x6520
DC.W 0x616E,0x6420,0x7374,0x7269,0x6B65,0x2061,0x6E79,0x206B
DC.W 0x6579,0x2077,0x6865,0x6E20,0x7265,0x6164,0x790D,0x0A00
DC.W 0,0,0,0,0,0,0,0
DC.W 0,0,0,0,0,0,0,0
DC.W 0,0,0,0,0,0,0,0
DC.W 0,0,0,0,0,0,0,0
DC.W 0,0,0,0,0,0,0,0
DC.W 0,0,0,0,0,0,0,0
DC.W 0,0,0,0,0,0,0,0
DC.W 0,0,0,0,0,0,0,0
DC.W 0,0,0,0,0,0,0,0x55AA
}

/*
 *  IBM style boot sector for nonsystem 3.5" disk:
 */
extern boot3();
asm{
boot3:
DC.W 0xEB34,0x9049,0x424D,0x2020,0x332E,0x3300,0x0202,0x0100
DC.W 0x0270,0x00A0,0x05F9,0x0300,0x0900,0x0200,0x0000,0x0000
DC.W 0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0012
DC.W 0x0000,0x0000,0x0100,0xFA33,0xC08E,0xD0BC,0x007C,0x1607
DC.W 0xBB78,0x0036,0xC537,0x1E56,0x1653,0xBF2B,0x7CB9,0x0B00
DC.W 0xFCAC,0x2680,0x3D00,0x7403,0x268A,0x05AA,0x8AC4,0xE2F1
DC.W 0x061F,0x8947,0x02C7,0x072B,0x7CFB,0xCD13,0x7267,0xA010
DC.W 0x7C98,0xF726,0x167C,0x0306,0x1C7C,0x0306,0x0E7C,0xA33F
DC.W 0x7CA3,0x377C,0xB820,0x00F7,0x2611,0x7C8B,0x1E0B,0x7C03
DC.W 0xC348,0xF7F3,0x0106,0x377C,0xBB00,0x05A1,0x3F7C,0xE89F
DC.W 0x00B8,0x0102,0xE8B3,0x0072,0x198B,0xFBB9,0x0B00,0xBED6
DC.W 0x7DF3,0xA675,0x0D8D,0x7F20,0xBEE1,0x7DB9,0x0B00,0xF3A6
DC.W 0x7418,0xBE77,0x7DE8,0x6A00,0x32E4,0xCD16,0x5E1F,0x8F04
DC.W 0x8F44,0x02CD,0x19BE,0xC07D,0xEBEB,0xA11C,0x0533,0xD2F7
DC.W 0x360B,0x7CFE,0xC0A2,0x3C7C,0xA137,0x7CA3,0x3D7C,0xBB00
DC.W 0x07A1,0x377C,0xE849,0x00A1,0x187C,0x2A06,0x3B7C,0x4038
DC.W 0x063C,0x7C73,0x03A0,0x3C7C,0x50E8,0x4E00,0x5872,0xC628
DC.W 0x063C,0x7C74,0x0C01,0x0637,0x7CF7,0x260B,0x7C03,0xD8EB
```

```
DC.W 0xD08A,0x2E15,0x7C8A,0x16FD,0x7D8B,0x1E3D,0x7CEA,0x0000
DC.W 0x7000,0xAC0A,0xC074,0x22B4,0x0EBB,0x0700,0xCD10,0xEBF2
DC.W 0x33D2,0xF736,0x187C,0xFEC2,0x8816,0x3B7C,0x33D2,0xF736
DC.W 0x1A7C,0x8816,0x2A7C,0xA339,0x7CC3,0xB402,0x8B16,0x397C
DC.W 0xB106,0xD2E6,0x0A36,0x3B7C,0x8BCA,0x86E9,0x8A16,0xFD7D
DC.W 0x8A36,0x2A7C,0xCD13,0xC30D,0x0A4E,0x6F6E,0x2D53,0x7973
DC.W 0x7465,0x6D20,0x6469,0x736B,0x206F,0x7220,0x6469,0x736B
DC.W 0x2065,0x7272,0x6F72,0x0D0A,0x5265,0x706C,0x6163,0x6520
DC.W 0x616E,0x6420,0x7374,0x7269,0x6B65,0x2061,0x6E79,0x206B
DC.W 0x6579,0x2077,0x6865,0x6E20,0x7265,0x6164,0x790D,0x0A00
DC.W 0x0D0A,0x4469,0x736B,0x2042,0x6F6F,0x7420,0x6661,0x696C
DC.W 0x7572,0x650D,0x0A00,0x4942,0x4D42,0x494F,0x2020,0x434F
DC.W 0x4D49,0x424D,0x444F,0x5320,0x2043,0x4F4D,0x0000,0x0000
DC.W 0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x55AA
}

WORD buf[5000];

main()
{
     register WORD *p;
     int  i, j, virgin, five, tracks, drive;
     long a, b, magic;

     Cconws("\033E\r\n\n\tMS-DOS compatible disk formatter, v1.1");
     Cconws("\r\n\n\tEnter size of disk (3/5): ");
     i = Cnecin();
     if (i == '5') {
          five = 1;
          tracks = 40;
          drive = 1;
     } else {
          five = 0;
          tracks = 80;
          drive = 0;
     }

     /* format all tracks */

     if (five)
       Cconws("\r\n\n\tAbout to  >> FORMAT <<  5.25\" disk in drive B.");
     else
       Cconws("\r\n\n\tAbout to  >> FORMAT <<  3.5\" disk in drive A.");
     Cconws("\r\n\n\t\tProceed (y/n)? ");
     i = Cnecin();
     if (i!='y' && i!='Y')  exit(0);
     Cconws("\r\n\n\tFormatting");
     magic = 0x87654321;
     virgin = 0xE5E5;
     for (i=0; i<tracks; i++) {       /* tracks*/
          if (five || (i&1))
               Cconout('.');
          for (j=0; j<2; j++) {       /* sides */
```

```
            if (Flopfmt(buf,0L,drive,9,i,j,1,magic,virgin)) {
                 Cconws("\r\n\nError formatting track ");
                 Cconout((i/10) + '0');
                 Cconout((i%10) + '0');
                 Cconws(" (decimal) on side ");
                 Cconout(j+'0');
                 Cconws("!\r\n");
                 goto bye;
            }
        }
    }

    /* Boot sector, FAT and root directory */

    for (i=0; i<2304; i++)      /* one track       */
        buf[i] = 0;
    if (five)
        j = 2;
    else
        j = 9;
    a = Flopwr(buf, 0L, drive, 1, 0, 1, j);
    if (five) {
        buf[256] = 0xFDFF; /* start of first FAT  */
        buf[257] = 0xFF00;
        buf[768] = 0xFDFF; /* start of second FAT */
        buf[769] = 0xFF00;
        p = (WORD *) &boot5;
    } else {
        buf[256]  = 0xF9FF;
        buf[257]  = 0xFF00;
        buf[1536] = 0xF9FF;
        buf[1537] = 0xFF00;
        p = (WORD *) &boot3;
    }
    for (i=0; i<256; i++)       /* boot sector */
        buf[i] = *p++;
    magic = Random()<<8;        /* get a random ID number */
    p = (WORD *) &magic;
    buf[4] = *p++;
    buf[5] = *p;
    b = Flopwr(buf, 0L, drive, 1, 0, 0, 9);
    if (a || b)
        Cconws("\r\n\n\tError writing root sectors! ");
    else
        Cconws("\r\n\n\tDone.  Hit any key.");
bye:
    Cnecin();
}


From 1512454875385577912
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,bc9ee608e503c61a,start
```

X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1987-11-29 09:50:46 PST
Relay-Version: version B 2.10 5/3/83; site utzoo.UUCP
Path: utzoo!mnetor!uunet!husc6!bbn!uwmcsd1!ig!jade!ucbvax!hplabs!decwrl!sgi!daisy!
atari-sources-request
From: atari-sources-request@daisy.UUCP
Newsgroups: comp.sources.atari.st
Subject: v00INF1: :: HOWTO - how to use the moderated atari groups
Message-ID: <678@daisy.UUCP>
Date: Sun, 29-Nov-87 12:50:46 EST
Article-I.D.: daisy.678
Posted: Sun Nov 29 12:50:46 1987
Date-Received: Wed, 2-Dec-87 21:12:27 EST
Sender: turner@daisy.UUCP
Lines: 181
Approved: daisy!turner

Submitted by: turner (D'arc Angel)
comp.sources.atari.st: Volume 0, Info 1
Archive-name: howto.use

Introduction:

This document outlines the purpose of comp.sources.atari.st and comp.binaries.atari.st; how to 'unpack' postings; and how to submit files to the groups for posting. These groups are moderated by me in my copious (:-) spare time and turn around may be a little slow. Don't be afraid to send me mail asking me what the ^$%# happened to your submission. Every program submitted to me is tested to the best of my abilities and hardware. If I can not fully test a submission, because I don't have appropriate hardware or I don't know the language being used, I will put a caveat in the posting warning the user to that effect.

Purpose:

The moderated groups were created to lessen the traffic created by multiple postings (as in: I lost part n, could someone repost it) and to provide a relatively reliable posting. To that end I download every posting; compile it (if appropriate); test it; document it (if necessary); checksum it; repackage it; locally post it; unpackage it and recheck the posting; and finally post it to the net. This process is fairly time consumming and therefore frustrating to the poster who would (very reasonably) like to see their posting see the light of day as soon as possible; have patience !

Contents:

Each posting consists of the following parts:

- Subject line with usage information
- Editor's note

checksums
the posting

1. Subject line

The subject line consists of a 3 field code which gives some information on the contents of the posting and a quick summary of what is in the posting. The 3 fields are seperated by ':'s and may in turn contain subfields seperated by ';'s. The first field is a 3 digit code that describes the language in which the posting is written. Some examples are:

bin - the posting is a binary (executable) only
mwc - the posting is written in Marc Williams C
mmc - Megamax C
drc - DRI (Alcyon) C
bas - basic
pps - Personal pascal
dra - Alcyon assembler
mwa - Marc Williams assembler
mad - MadCap assembler
pro - prolog
doc - documentation file (no executables)

(I will add more as the need arises)

The second field contains the resolutions in which the program will run, if it can run in more than 1 resolution, then they seperated by ';'s. The values are:

all - all resolutions
b&w - monochrome
low - low res color
med - hi res color

The third field lists the filters that need to be run to 'unpackage' the posting, that is get it into a format that can be used on the ST. If more than one filter needs to be used they are seperated by ';'s and listed in the order that they must be executed. The values are:

shar - Un*x Bourne shell
uue - uudecode
arc - arc
     - (empty) posting is usable as is

for example, uue;arc would mean that you must first run the uudecode program on the posting, then run the arc program on the output of the uudecode program.

Editor's Note:

This is a (hopefully) quick note from me explaining what the posting is about, any gotcha's that I found in using it, etc.

Checksums:

These are the checksums for each part (if more than one) of the complete posting

How to 'unpackage' postings:

The 2 most popular methods for packaging posting are shar format for sources and uue;arc for binaries. To unpack a shar format simpley remove the leading headers (usually marked by a line stating 'run sh on everything that follows...') and trailing signatures and run the stripped file thru the Un*x sh command (sh filename). This will unpack and check each file in the shar. If you are not running Un*X then you need to manually seperate the parts with an editor (what a pain).
To unpackage a uue;arc submission execute the following steps:

1. look at the line 'begin part ...' in the first (or only) part and save the file as that filename with the file extension '.uue' you need not change the file as uudecode ignores everything before the 'table' line. If the posting has more than 1 part name then the file extension to part 1 should be '.uaa' and the filename for part 2 'filename.uab', part 3 'filename.uac' etc. For example a 3 part posting of the gulaam.ttp program would be saved as:

gulaam.uaa
gulaam.uab
gulaam.uac

It should be noted that this is for the 'official' uudecode program from John-Pierre Dumas that is posted periodically to the group comp.binaries.atari.st. Other versions of uudecode may be used but their naming conventions may be different. The other most popular form of file naming would name the above 3 parts:

gulaaa.uue
gulaab.uue
gulaac.uue

In general the naming convention is only a formality, the basic format of a uue file is understood by all flavours of uudecode.

2. type 'uudecode filename.uue' for single part postings or 'uudecode filename.uaa' for multiple parts, the other parts will be automatically linked in.

3. you should now have a file named 'filename.arc' (gulaam.arc in our example), type 'arc x filename' to extract the pieces of the original posting.

4. that's it.

The chicken or the egg:

One major problem has been how to post the uuen/decode binaries without putting them into a uue format. To that end a uudecode.bas program is periodically posted to the net, this is a 'bootstrap' version which will decode the uuencode... (you get the idea). This program does NOT understand anything before the 'begin' line or after the 'end' statement.

How to submit files:

mail submissions to:

....{decwrl,ucbvax}!imagen!atari!turner

PLEASE include documentation as to what the file contains, how to use it, etc. If it is a binary file, remember to uuencode it first as most mailer can't handle binary (non-ascii) characters.

Toolkit:

The full set of programs used in decoding files from the moderated groups are:

```
uudecode.bas - basic program used to decode the real uudecode.ttp
uudecode.ttp - decode encoding postings
uuencode.ttp - encode binaries for posting
arc.ttp      - archive/dearchive groups of binaries files
cksm.ttp     - checksum a file
```

Archives:

At this time, all files posted to both moderated groups are also mailed to ...umix!hyc (Howard Chu) who maintains an archive available to both usenet and ARPAnet. Contact him for more details.

From -2508408087102301065
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,a8bd1f65ecb5cec0,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1987-12-12 17:33:01 PST
Path: utzoo!utgpu!water!watmath!clyde!cbosgd!ucbvax!ucbcad!ames!sgi!daisy!atari-sources-request
From: atari-sources-request@daisy.UUCP
Newsgroups: comp.sources.atari.st
Subject: v00i049: mwc:all: GULAMFIX - keyboard mod program for gulam shell
Message-ID: <760@daisy.UUCP>
Date: 13 Dec 87 01:33:01 GMT

Sender: turner@daisy.UUCP
Lines: 367
Approved: daisy!turner

Submitted by: sun!mandrill!pm (Prabhaker Mateti)
comp.sources.atari.st: Volume 0, Issue 49
Archive-name: gulamfix1

~+~+~+

```
#define F4    0204
#define F5    0205
#define F6      0206
#define F7    0207
#define F8    0210
#define F9    0211
#define F10   0212

#define HELP    0213
#define UNDO  0214
#define INSERT    0215
#define HOME  0216

#define UPARRO    0273
#define DNARRO    0274
#define LTARRO    0275
#define RTARRO    0276

#define  KLP  0300
#define  KRP  0301
#define  KSTAR    0302
#define  KPLUS    0303
#define  KENTER   0304
#define  KMINUS   0305
#define  KDOT 0306
#define  KSLASH   0307

#define  K0   0260
#define  K1   0261
#define  K2   0262
#define  K3   0263
#define  K4   0264
#define  K5   0265
#define  K6   0266
#define  K7   0267
#define  K8   0270
#define  K9   0271


/* The above are the unshifted values.  Shifted, as well as
'capslocked' (for only these keys) values that I use are these +0040.
*/

#define SHIFTED 0040

/* The distributed beta-version 1.00.00.02 has the following key map.
Edit it to reflect your taste/needs */

     static
char keymap[128*3] = {
     /* -------scan codes: 0x00-0x0f ---------------------------- */
     '\000' /*nul*/, '\033' /*esc*/,
     '\061' /*1*/, '\062' /*2*/, '\063' /*3*/,
```

```
'\064' /*4*/, '\065' /*5*/, '\066' /*6*/, '\067' /*7*/,
'\070' /*8*/, '\071' /*9*/, '\060' /*0*/,
'\055' /*-*/, '\075' /*=*/,
'\010' /* bs*/, '\011' /* ht*/,

/* ------------------ 0x10-0x1f ---------------------------- */
'\161' /*q*/, '\167' /*w*/, '\145' /*e*/, '\162' /*r*/,
'\164' /*t*/, '\171' /*y*/, '\165' /*u*/, '\151' /*i*/,
'\157' /*o*/, '\160' /*p*/,
'\133' /*[*/, '\135' /*]*/,
'\015' /* cr*/, '\000' /*ctrl*/,
'\141' /*a*/, '\163' /*s*/,

/* ------------------ 0x20-0x2f ---------------------------- */
'\144' /*d*/, '\146' /*f*/, '\147' /*g*/, '\150' /*h*/,
'\152' /*j*/, '\153' /*k*/, '\154' /*l*/,
'\073' /*;*/, '\047' /*'*/, '\140' /*`*/,
'\000' /*lshift*/,
'\134' /*\*/,
'\172' /*z*/, '\170' /*x*/, '\143' /*c*/,  '\166' /*v*/,

/* ------------------ 0x30-0x3f ---------------------------- */
'\142' /*b*/, '\156' /*n*/, '\155' /*m*/,
'\054' /*,*/, '\056' /*.*/, '\057' /* / */,
'\000' /*rshift*/, '\000' /*unused*/, '\000' /*alt*/,
'\040' /*space*/,
'\000' /*caps lock*/,
F1, F2, F3, F4, F5,


/* ------------------ 0x40-0x4f ---------------------------- */
F6, F7, F8, F9, F10,
'\000' /*unused*/, '\000' /*unused*/,
HOME,    UPARRO,   '\000' /*unused*/,
KMINUS,  LTARRO,   '\000' /*unused*/,
RTARRO,  KPLUS,    '\000' /*unused*/,

/* ------------------ 0x50-0x5f ---------------------------- */
DNARRO,  '\000' /*unused*/, INSERT,  '\177' /*del*/,
'\000' /*unused*/,'\000' /*unused*/,'\000' /*unused*/,
'\000' /*unused*/,'\000' /*unused*/,'\000' /*unused*/,
'\000' /*unused*/,'\000' /*unused*/,'\000' /*unused*/,
'\000' /*unused*/,'\000' /*unused*/,'\000' /*unused*/,

/* ------------------ 0x60-0x6f ---------------------------- */
'\000' /*ISO key*/,
UNDO,    HELP,    KLP, KRP, KSLASH,  KSTAR,
K7,  K8,  K9,  K4,  K5,  K6,
K1,  K2,  K3,

/* ------------------ 0x70-0x72 ---------------------------- */
K0,  KDOT,    KENTER,
```

```
    /* ------------------ 0x73-0x7f  undefined ----------------- */
    '\000','\000','\000','\000','\000','\000','\000','\000',
    '\000','\000','\000','\000','\000',
/**
 };


static
char shif[128] = {
**/
    /* ------------------ 0x00-0x0f ---------------------------- */
    '\000' /*nul*/, '\033' /*esc*/,
    '\041' /*!*/, '\100' /*@*/,  '\043' /*#*/, '\044' /*$*/,
    '\045' /*%*/, '\136' /*^*/, '\046' /*&*/, '\052' /***/,
    '\050' /*(*/, '\051' /*)*/, '\137' /*_*/, '\053' /*+*/,
    '\010' /* bs*/, '\011' /* ht*/,

    /* ------------------ 0x10-0x1f ---------------------------- */
    '\121' /*Q*/, '\127' /*W*/, '\105' /*E*/, '\122' /*R*/,
    '\124' /*T*/, '\131' /*Y*/, '\125' /*U*/, '\111' /*I*/,
    '\117' /*O*/, '\120' /*P*/,
    '\173' /*{*/, '\175' /*}*/,
    '\015' /* cr*/, '\000' /*ctrl*/,
    '\101' /*A*/, '\123' /*S*/,

    /* ------------------ 0x20-0x2f ---------------------------- */
    '\104' /*D*/, '\106' /*F*/, '\107' /*G*/, '\110' /*H*/,
    '\112' /*J*/, '\113' /*K*/, '\114' /*L*/,
    '\072' /*:*/, '\042' /*"*/, '\176' /*~*/,
    '\000' /*lshift*/,
    '\174' /*|*/,
    '\132' /*Z*/, '\130' /*X*/, '\103' /*C*/,  '\126' /*V*/,

    /* ------------------ 0x30-0x3f ---------------------------- */
    '\102' /*B*/, '\116' /*N*/, '\115' /*M*/,
    '\074' /*<*/, '\076' /*.*/, '\077' /* ? */,
    '\000' /*rshift*/, '\000' /*unused*/, '\000' /*alt*/,
    '\040' /*space*/,
    '\000' /*caps lock*/,
    F1+SHIFTED, F2+SHIFTED, F3+SHIFTED, F4+SHIFTED, F5+SHIFTED,


    /* ------------------ 0x40-0x4f ---------------------------- */
    F6+SHIFTED, F7+SHIFTED, F8+SHIFTED, F9+SHIFTED, F10+SHIFTED,
    '\000' /*unused*/, '\000' /*unused*/,
    HOME+SHIFTED, UPARRO+SHIFTED,    '\000' /*unused*/,
    KMINUS+SHIFTED,LTARRO+SHIFTED,   '\000' /*unused*/,
    RTARRO+SHIFTED,KPLUS+SHIFTED,    '\000' /*unused*/,

    /* ------------------ 0x50-0x5f ---------------------------- */
    DNARRO+SHIFTED,    '\000' /*unused*/,
    INSERT+SHIFTED,    '\177' /*del*/,
    '\000' /*unused*/,'\000' /*unused*/,'\000' /*unused*/,
```

```
    '\000' /*unused*/,'\000' /*unused*/,'\000' /*unused*/,
    '\000' /*unused*/,'\000' /*unused*/,'\000' /*unused*/,
    '\000' /*unused*/,'\000' /*unused*/,'\000' /*unused*/,

    /* ------------------ 0x60-0x6f ---------------------------- */
    '\000' /*ISO key*/,
    UNDO+SHIFTED, HELP+SHIFTED, KLP+SHIFTED,
    KRP+SHIFTED,  KSLASH+SHIFTED,KSTAR+SHIFTED,
    K7+SHIFTED,   K8+SHIFTED,   K9+SHIFTED,
    K4+SHIFTED,   K5+SHIFTED,   K6+SHIFTED,
    K1+SHIFTED,   K2+SHIFTED,   K3+SHIFTED,

    /* ------------------ 0x70-0x72 ---------------------------- */
    K0+SHIFTED,   KDOT+SHIFTED, KENTER+SHIFTED,

    /* ------------------ 0x73-0x7f  undefined ----------------- */
    '\000','\000','\000','\000','\000','\000','\000','\000',
    '\000','\000','\000','\000','\000',
/**
};

static
char caps[128] = {
**/
    /* ------------------ 0x00-0x0f ---------------------------- */
    '\000' /*nul*/, '\033' /*esc*/,
    '\061' /*1*/, '\062' /*2*/, '\063' /*3*/,
    '\064' /*4*/, '\065' /*5*/, '\066' /*6*/, '\067' /*7*/,
    '\070' /*8*/, '\071' /*9*/, '\060' /*0*/,
    '\055' /*-*/, '\075' /*=*/,
    '\010' /* bs*/, '\011' /* ht*/,

    /* ------------------ 0x10-0x1f ---------------------------- */
    '\121' /*Q*/, '\127' /*W*/, '\105' /*E*/, '\122' /*R*/,
    '\124' /*T*/, '\131' /*Y*/, '\125' /*U*/, '\111' /*I*/,
    '\117' /*O*/, '\120' /*P*/,
    '\133' /*[*/, '\135' /*]*/,
    '\015' /* cr*/, '\000' /*ctrl*/,
    '\101' /*A*/, '\123' /*S*/,

    /* ------------------ 0x20-0x2f ---------------------------- */
    '\104' /*D*/, '\106' /*F*/, '\107' /*G*/, '\110' /*H*/,
    '\112' /*J*/, '\113' /*K*/, '\114' /*L*/,
    '\073' /*;*/, '\047' /*'*/, '\140' /*`*/,
    '\000' /*lshift*/,
    '\134' /*\*/,
    '\132' /*Z*/, '\130' /*X*/, '\103' /*C*/,  '\126' /*V*/,

    /* ------------------ 0x30-0x3f ---------------------------- */
    '\102' /*B*/, '\116' /*N*/, '\115' /*M*/,
    '\054' /*,*/, '\056' /*.*/, '\057' /* / */,
    '\000' /*rshift*/, '\000' /*unused*/, '\000' /*alt*/,
    '\040' /*space*/,
```

```
     '\000' /*caps lock*/,
     F1, F2, F3, F4, F5,


     /* ------------------ 0x40-0x4f ---------------------------- */
     F6, F7, F8, F9, F10,
     '\000' /*unused*/, '\000' /*unused*/,
     HOME,    UPARRO,   '\000' /*unused*/,
     KMINUS,  LTARRO,   '\000' /*unused*/,
     RTARRO,  KPLUS,    '\000' /*unused*/,

     /* ------------------ 0x50-0x5f ---------------------------- */
     DNARRO,   '\000' /*unused*/, INSERT,   '\177' /*del*/,
     '\000' /*unused*/,'\000' /*unused*/,'\000' /*unused*/,
     '\000' /*unused*/,'\000' /*unused*/,'\000' /*unused*/,
     '\000' /*unused*/,'\000' /*unused*/,'\000' /*unused*/,
     '\000' /*unused*/,'\000' /*unused*/,'\000' /*unused*/,

     /* ------------------ 0x60-0x6f ---------------------------- */
     '\000' /*ISO key*/,
     UNDO,    HELP,    KLP, KRP, KSLASH,  KSTAR,
     K7,  K8,  K9,  K4,  K5,  K6,
     K1,  K2,  K3,

     /* ------------------ 0x70-0x72 ---------------------------- */
     K0,  KDOT,    KENTER,

     /* ------------------ 0x73-0x7f  undefined ----------------- */
     '\000','\000','\000','\000','\000','\000','\000','\000',
     '\000','\000','\000','\000','\000'

};

#include <osbind.h>

/* Return offset in the binary file gulam.prg where the keymap[] begins.
74844L is this number valid for beta-version 1.00.00.02 */

     char *
kmpbgn(buf, n)
register char *buf;
register long n;
{
     /* kmp[] is extracted from Gulam's src; do NOT change it. */
     static char   kmp[16] = {
          /* -------scan codes: 0x00-0x0f --------------- */
          '\000' /*nul*/, '\033' /*esc*/,
          '\061' /*1*/, '\062' /*2*/, '\063' /*3*/,
          '\064' /*4*/, '\065' /*5*/, '\066' /*6*/, '\067' /*7*/,
          '\070' /*8*/, '\071' /*9*/, '\060' /*0*/,
          '\055' /*-*/, '\075' /*=*/,
          '\010' /* bs*/, '\011' /* ht*/
     };
```

```
     register long i;
     register int  j;
     register char *p, *q;

     for (i = 0L; i < n; i++)
     {    p = &buf[i]; q = kmp;
          for (j = 0; j < 16; j++)
               if (*p++ != *q++) break;
          if (j == 16) goto found;
     }
     printf("key map bgn not found! aborting...\n");
     exit(-1);

     found:   printf("key map begins at byte %ld\r\n", i);
     return   &buf[i];
}


main()
{
     register char *buf, *p, *q;
     register int  i;
     register long n, m;
     register unsigned long lsz;

     lsz  = 90000L;
     buf = (char *) Malloc(lsz); if (buf == 0L) goto error;

     /* read all of gulam.prg into buf */
     i = Fopen("gulam.prg", 0);
     if (i <= 0) {printf("gulam.prg not found\n"); goto error;}
     n =  Fread(i, lsz, buf);    if (n <= 0L) goto error;
        Fclose(i);

     /* fix the 128*3 bytes of keymap[] */
     p = keymap; q = kmpbgn(buf, n);
     for (i=0; i < 128*3; i++) *q++ = *p++;

     /* write the new gulam.prg out */
     i = Fcreate("gulamnew.prg", 0);
     if (i <= 0)
     {error:
          printf("** could not create gulamnew.prg **\n");
          exit(-2);
     }
     printf("new version of gulam is gulamnew.prg ");
     m =  Fwrite(i, n, buf);if (m != n) goto error;
        Fclose(i);
     printf("\n");
}

/* -eof- */
```

From 623197522000406391
X-Google-Language: ENGLISH,ASCII-7-bit
X-Google-Thread: 100493,e4ef338352e7580e,start
X-Google-Attributes: gid100493,public
X-Google-ArrivalTime: 1987-12-12 17:46:55 PST
Path: utzoo!utgpu!water!watmath!clyde!cbosgd!ucbvax!ucbcad!ames!sgi!daisy!atari-sources-request
From: atari-sources-request@daisy.UUCP
Newsgroups: comp.sources.atari.st
Subject: v00i050: :: TERMCAP termcap entry for curses program
Message-ID: <761@daisy.UUCP>
Date: 13 Dec 87 01:46:55 GMT
Sender: turner@daisy.UUCP
Lines: 28
Approved: daisy!turner

Submitted by: turner (D'arc Angel)
comp.sources.atari.st: Volume 0, Issue 50
Archive-name: termcap

+~+~+~